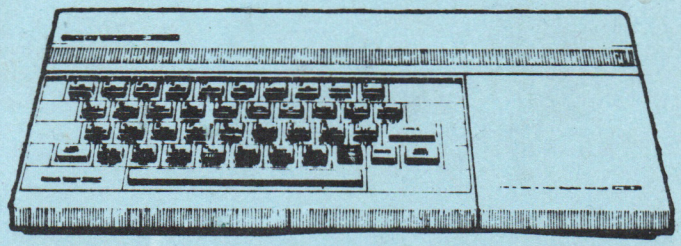
MSCRIPT

WORD PROCESSOR

USERS MANUAL

BY
KARL H. MEYER

MICRO–SYSTEMS SOFTWARE,INC.



PUBLISHED FOR THE
TIMEX 2068 COMPUTER
BY ZEBRA SYSTEMS,INC.

Serial
1174

Serial
1174

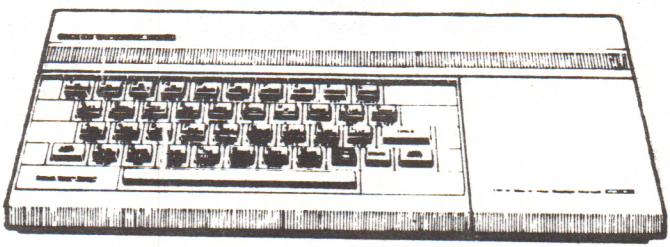
MSCRIPT

WORD PROCESSOR

USERS MANUAL

BY
KARL H. MEYER

MICRO–SYSTEMS SOFTWARE,INC.



PUBLISHED FOR THE
TIMEX 2068 COMPUTER
BY ZEBRA SYSTEMS,INC.
UNDER LICENSE FROM
MICRO–SYSTEMS SOFTWARE,INC.

MSCRIPT WORDPROCESSOR
for the
TIMEX 2068 COMPUTER

Congratulations on selecting MSCRIPT for your Timex/Sinclair 2068 computer! Because the MSCRIPT User's Manual is written for a wide variety of computers, you need to read these few pages before you begin to use it.

MSCRIPT is supplied on cassette, with side one containing the MSCRIPT software, and side two containing the sample files. We advise you to immediately make a backup copy using the Backup feature of the loader to make a "Working Copy". Put your master tape in a safe place, and always use your working copy. This assures that in case of damage to your working copy tape you can make a new one from your master, without having to send us the tape for replacement.

Starting MSCRIPT

Make sure your printer interface is attached to the expansion connector at the rear of your TS2068. Place your tape in your tape player and type:

LOAD "MSCRIPT" or LOAD ""

Press the play button on your tape player. When MSCRIPT is loaded you will see the MSCRIPT Loader Menu. From this menu you may select your Printer Interface, make a customized Backup Copy, or execute MSCRIPT.

Special Keys for MSCRIPT

MSCRIPT makes use of a number of keys on your keyboard for special purposes. These keys are defined as follows:

The Function Key

The combination of the CAPS SHIFT and the SYMBOL SHIFT keys are what we call the function key, and it is used as a control key. It is designated in this way: <funct>. If the manual states "Press <funct>-H" it means you are to hold down the CAPS SHIFT and the SYMBOL SHIFT keys and press the letter H at the same time.

The Command Key

The combination of the CAPS SHIFT and the BREAK key is called the command key, which is designated as <command>. This works as a toggle switch to take you to and from the TEXT ENTRY/EDIT SCREEN to the COMMAND MENU and back again. It will also be used to abort the operation that is in progress: if you are printing, it will cause the printer to stop; if you mistakenly go to the Print Menu, press <command> and it returns you to the TEXT ENTRY/EDIT SCREEN.

Cursor Movement

The four arrow keys are used to move your cursor through the text in the directions indicated. Pressing the CAPS SHIFT and the appropriate ARROW KEY will move the cursor in the desired direction.

Backing Up Your Work

The importance of making backups of all your important documents cannot be stressed too strongly. One method is to make multiple copies of a text file on the same tape, another method is to have 2 cassette tapes and save the file once on each tape. We also recommend that you save the document frequently during composition so that in case of a power outage, or other disruption you will have a copy with only the very latest changes and additions lost. Remember... A cassette tape costs $1.00, how much does 8 hours of typing cost?

CONTENTS

Introduction 1

Part I – What Is A Word Processor? 5

An overview of Word Processing and how the word processor has revolutionized the modern office.

Part II – MSCRIPT In Use 13

Learning to use all the commands that are available with MSCRIPT.

Part III – Some Special Applications 43

How to use MSCRIPT for keeping track of notes, stamp collections, and other things; mailing lists, letter–heads and evelopes; page oriented editing for BASIC programmers.

Glossary 55

Almost everything you wanted to know about the terms associated with word processing.

Timex 2068 MSCRIPT Addenda 71

Using MSCRIPT with MTERM 72

Printer Codes and other Tricks 73

Index I

Introduction

One of the most widely used, misused and abused terms in the computer industry is "user friendly." In the Glossary at the end of this manual it is defined as, "Software and hardware systems that are easy for the operator to learn and use without a long period of training." In this sense, MSCRIPT is user friendly, for most typists can learn how to enter text and perform basic editing functions in a matter of minutes. This manual was written with the newcomer in mind, and is as free of "computerese" as I have been able to make it. Part I is a look at word processing in general, and shows you how modern word processing has changed the way we work. The few terms that are absolutely essential are defined for you, and from there we show you how MSCRIPT works.

Unlike other word processors, MSCRIPT is available in a version for almost all microcomputers, and comes ready to run on your machine. Whether you have a Timex-Sinclair or an IBM-PC, your version is ready use without lengthy and complicated installation, or customizing. MSCRIPT is ready to run.

Before you do anything else, read the instructions for your particular computer which are on separate sheets packed with this manual. Because of the differences in keyboards, not all machines use the same key to perform the same thing. Two of the keys on your keyboard have been designated the Function Key and the Command Key. These two keys control all the functions of this program.

I have used many word processors over the past five years on a wide variety of computers. As a writer, there are many features I find essential; others really sound good, but I find that I have little or no use for them. These are what I call the bells and whistles, the sizzle that sells the steak. I bought two enhancements for the word processor I'd used for two years, because they sounded useful. How useful? One turned out to be more work than I care for, and the other has never worked properly, despite having two "upgrades" and "patches" from the author. The price of those two utilities alone was greater than the cost of MSCRIPT, and they're both included in MSCRIPT. And they both work properly and with an ease that makes me wish I'd had MSCRIPT three years ago.

In addition to being a writer (novels, articles, software reviews), I have taught word processing at a business college,

an adult education center, and for one of the largest corporations in America. As a part-time consultant, I do some on-site training for individuals and small businesses who are getting into word processing for the first time. My wife is also a teacher of word processing at two community continuing education centers. Between us we have trained close to a thousand people in how to use word processors for almost every known application. These people have been secretaries, high school students, professional writers, doctors, lawyers and businessmen of all kinds, and each has different needs and desires when selecting a word processor.

Many people have taken a common step in upgrading to a more powerful computer, or simply one they like better. I've spoken with some who have said, "Well, I'd like to get rid of those three Zanzibars and put in WonderComps, but it took so long to get those girls trained in using the word processor that I don't want to have to go through a long period of retraining." With MSCRIPT this problem is gone. MSCRIPT on a TRS-80 computer is the same program as it is on the IBM, the Sanyo, or any CP/M machine. The only difference is in the few keys that are assigned special functions. MSCRIPT is truly a universal word processor that can move with you from system to system as you upgrade to newer computers without relearning.

Because of this portability, or compatability with so many machines, I found it was necessary to do a little "fudging" in some of the terminology of this manual. You'll find several instances where I state, "... you will see something like this on your screen." If you have an 80-column monitor, obviously your screen will appear a bit different than a 64-column, or a 40-, or even a 20-column. For simplicity, I've used the following method of indicating three keys:

<funct> stands for the Function Key
<command> stands for the Command Key
<enter> stands for the Enter Key, Return Key, Carriage Return, or the key with the arrow pointing down and to the left.

Throughout the manual, the word "Press" means to press the key indicated. In the case of the Function Key, you'll see the following: "Press <funct>-H," and it means to hold down the Function Key and press the H. The word "Enter" means you are to type in something and then press <enter>.

* * *

Three years ago Micro-Systems Software had a suggested schedule for development of a number of programs, and an unnamed word processor was included. I first met Mark Lautenschlager about two years ago. Mark is General Manager of MSS, and we have become good friends. We talked word processors in general and word processors in particular. He was constantly asking me, "What do you like about this package?" or "Why don't you take a look at this one?" I didn't realize at the time he was picking my brain to find out what I wanted in a word processor, what features I considered essential, and what features I can live without. I gave him honest answers about my preferences, as well as those I had gleaned over the years in teaching word processing.

When it was time to write MSCRIPT, Mark passed those comments along to Steve Pagliarulo, the author of the program, and when they asked me to write the manual, I was pleased to find nearly everything I wanted in a word processor was there. I began writing the manual with a pre-release version of MSCRIPT, and as a field tester was able to find only one bug in the program. I tried everything I knew to make the program hang up or crash, but was totally unsuccessful. The only time I thought I'd succeeded was the time it refused to print; after several attempts I learned my printer was connected to another computer! In the few instances I felt the program lacked something, a phone call or brief comment at lunch with Steve was all it took to have it implemented. It's rare to find that kind of generous cooperation in this industry.

Acknowledgments

When I began writing this manual, I enlisted the aid of a fellow writer, Allan DeLoach. I told him what the Function and Command Keys were, and asked him to see what he thought of the program. "What about the manual?" he asked.

"I haven't started writing it yet."

"Good. Then I won't have to wade through several hundred pages of stuff I can't understand."

Allan is a pharmacist and writer, and while he's used his computer for word processing for almost two years, he isn't interested in "bits, bytes and buffers." He's typical, for most people don't care about the hows and whys of computer technology, they just want to get some work done with it.

He worked with MSCRIPT for a week before receiving the first

portion of the manual, and then had to wait another week for the rest of the first draft. His suggestions and comments have helped tremendously in my writing of what I hope is a manual that can be understood by anyone, especially those without computer backgrounds.

Mark Lautenschlager has given me invaluable assistance with terminology and operational theory of microcomputers in general. His expertise with hardware and software has been available any time I ask.

Jeanne Kesmar and Melvin Weaver gave me some much appreciated feedback for the manual.

My wife Phyllis was, as always, encouraging and patient, even allowing me to tie up her computer for several days while I had MSCRIPT running on three computers in our office.

Most helpful of all, though, has been Steve Pagliarulo. Not once has he complained about the phone calls I've made to ask a question or make a suggestion, even though he was deeply involved in working with something that was more pressing at the moment. His generosity with his time and knowledge have made this manual a pleasure.

-- Karl H. Meyer

(This MSCRIPT User's Manual was written entirely with MSCRIPT on a Radio Shack Model III computer, and printed on a Radio Shack Daisy Wheel II printer.)

Part I
What Is A Word Processor?

A word processor can be thought of as a solution to problems that arise as a result of creating, correcting, filing, changing and printing words. When we speak of a "word processor," we refer to three distinct items that function together: The computer, the printer, and the software that turns the computer into a word processor. Your computer and printer become a word processor only with the addition of suitable software, like MSCRIPT.

Word processing has revolutionized the way we deal with putting words on paper. Consider the old-fashioned office. A typist is handed a memo or takes dictation for a letter, which is typed as a "rough draft." The rough draft is then examined by the person who dictated it, marked for any corrections or changes, and then returned to the typist. The "final draft" is then typed, incorporating all the indicated changes; any typographical errors are carefully corrected with lift-off tape or correction fluid and erasers, then it is waved in the air to dry, and the crumbs are carefully blown away. The final draft is once again routed to the originator for signature. It is not uncommon at this point for second thoughts to occur, with the following results:

"Miss Pelham, how about moving the third paragraph on page four to this place on page one?" The entire document is then retyped, corrections made, etc. At four-thirty the comment is made, "It's almost time to leave, and only five letters ready to mail? What have you been doing all day?"

Word processors have eliminated most of Miss Pelham's drudgery. Once the rough draft has been typed, corrections are made quickly and easily, without having to retype anything that is correct the first time. Overtyping is accomplished without the use of lift-off tapes or correction fluids (and how many hours did Miss Pelham lose every week waiting for it to dry, or diluting the gummy mass in the bottom of the bottle?). Words, letters, sentences, whole paragraphs or even additional documents can be inserted without having to retype the original. If Miss Pelham can't quite remember whether the word is "procedure" or "proceedure," she doesn't have to slow down to think each time, but may change all instances of a misspelled word later with a global search and change command. If she finds the name "Pelczarski" clumsy to type, and it will appear

frequently, she may type "ppp" whenever it occurs, and change them all to "Pelczarski" with the same command.

No longer is it necessary to store carbon copies of documents in bulky file cabinets. Permanent copies are stored on diskettes which occupy much less space, and printed copies can be obtained quickly and easily if needed.

Word processing has changed the way professional writers work too. Most writers take copious notes on scraps of paper, loose leaf binders, match book covers, business cards, paper napkins, or anything else convenient to hand. Prior to beginning to write a story, the notes must be sorted out, classified, indexed, or otherwise put in order to be of use. The same global search command that Miss Pelham uses to find the "ppp" can be used to locate any desired piece of information for the writer, once he has transferred the notes to a memo file. This command can also be used to advantage to create a powerful data base. (This is discussed fully in Part 3.)

At the end of this MSCRIPT User's Manual is a glossary of terms encountered in word processing. The terms that are absolutely essential for your understanding of how MSCRIPT works will be presented as we go.

The Document is the name given to any material you produce with MSCRIPT. It can be material that has been completed and stored on disk or tape, or the text you are working with currently. It can be a letter, report, memo, ad copy--in short, anything made up of words that you create, edit, or print with your word processor. It is an entire text file that is referenced by a single file name. LETTER1, REPORT05, LETTER2/J3 (or LETTER2.J3, depending on your Computer) are several examples of document names.

The Screen is what we call what you see on your monitor or your TV when used as part of your word processing system. When you first start MSCRIPT, you see the Text Entry/Edit Screen.

The Cursor is the graphic symbol on your screen showing you where you are working at the moment. Any character you type will appear at the cursor position.

A Menu is a list of choices available to you. MSCRIPT has three menus, the Command Menu, the Help Menu, and the Print Menu. Each of these will be examined in detail.

A Video Line is the line of text on your screen from its first character on the left to the last character on the right. A video line may be a sentence, a part of a sentence, or several

sentences.

A Character is the smallest amount of text recognized by MSCRIPT. It is a single letter, number, symbol, or space, and occupies one horizontal space on the screen. Even though you do not see spaces, MSCRIPT considers a space to be as important as the letters M and S. Examples of symbols are, *, /, ~, [,], {, and }.

A Word is any group of characters beginning and ending with a space or a tab." Word" is a word; so is "23{9_|^]."

A Sentence is any group of characters that ends with a period, question mark or exclamation point.

A Paragraph is any stream of text that ends when you press ENTER.

A Block is any amount of text surrounded by special graphic symbols representing Block Markers. A block can be as short as one character, or as long as the entire document.

To Update a document means to make permanent changes in a document that already exists. When it is saved to the disk under the same name, it is said to be updated.

Text Entry is the act of creating a document by typing the document from the keyboard.

Editing is revising the current document by adding more text, deleting, inserting, changing, etc.

The Buffer is an area of your computer's memory that serves as a work table for your documents. It is a temporary storage area.

To Load a document is to copy it from the disk to the buffer. This is similar to taking a file folder from a cabinet and placing it on a desk where it can be used.

To Save a document is to copy it from the buffer to the disk. This is similar to placing a file folder in the file cabinet when you are finished working with it.

(Loading and Saving documents, while similar to handling file folders, are vastly different in that only copies of the document are moved. This will be discussed fully later on.)

A String is a unique group of characters. In word processing, a string is most commonly used to refer to that

group of characters you wish to locate within a document ("find string"), or the group of characters you wish to replace it with ("change string").

The Window is the screen "opening" which allows you to view a portion of your document. Think of your document as a huge sheet of paper, wider and longer than your screen; the screen represents a "window" in a wall, allowing you to view a portion of the document behind the wall as it is shifted up, down and sideways.

Let's Get Started!

If you have not yet read and followed the directions in the Introduction about making backup copies of your MSCRIPT program, and installed it on a system disk, do that now, then return to this page. If you have more than one drive available, write protect your system disk, and have a formatted data disk in another drive. Please note that while we will make constant reference to disks (also called diskettes) in this manual, some versions of MSCRIPT are available for tape machines. When you are told to "save to disk," if you have a tape machine, you will save the document to tape. Be sure you have read the first page of the manual following the title page. This will give you all the specific instructions for your particular computer, such as defining which key on the keyboard is used for the Function Key and Command Key. It will also tell you how to make backup copies of your MSCRIPT disk or tape, and and other information that applies to your computer.

Turn on your machine, and at the ready prompt, type MSCRIPT and press <enter>. The first screen you see will be the Text Entry/Edit screen. At the bottom of the screen you see a row of hyphens and +'s, which is called the Tab Line, and serves as a measurement of your text width and cursor position. Immediately below the Tab Line is the Status Line, which provides the following information:

Window: 001,001 = Indicates your cursor is located on the first horizontal row of the screen, and in the first column from the left side. Note that this is the postion of the cursor on the screen, not the document itself.

Column: 001 = Indicates your cursor is in the first row on the left side of the document.

Line: 1 = Indicates the cursor is on the first line of your document.

FUNCT-H for Help! = A constant reminder that you may turn to the Help Menu at any time while you are working on your document.

Begin typing something (anything), and watch the numbers change, constantly keeping track of where your cursor is located. When you come to the end of a line, do not press <enter>, simply keep typing. Any word that is too long to fit on the current line will be moved to the next line. This is known as "wraparound", and is fast and automatic. There is no need to slow down at the end of a line to decide where to break a word, nor is it necessary to wait for the word to be "wrapped." You will not lose any characters in the process.

Notice that when you press <enter>, a graphic symbol is displayed on the screen, and the cursor goes to the beginning of the next line. This symbol will not appear in your printed text, but serves two purposes: It tells the printer that this is the end of this line, and to go to the next for further instructions. It also is an indication to you that you have terminated the line at that point. Do not make the mistake of filling the end of a line with spaces and then going to the next line to input more text. Remember that even though spaces do not show, the computer considers them characters the same as letters, numbers and symbols. If you change the format (dimensions) of your document, the extra spaces are going to be treated as any other string of text. (Any time you get weird looking printed copy, check to see if you have substituted spaces for a properly terminated line.)

Press <command>, and your screen now displays the Command Menu, which looks something like this:

```
MSCRIPT Command Menu

A - Append document                [A docname]
C - Change string                  [Cstring]
D - DOS commands/exit              [D command]
F - Find string                    [Fstring]
L - Load document from disk        [L docname]
R - Restart MSCRIPT                [R]
S - Save document                  [S docname]
T - Set tab positions              [T x,x,x,x]
W - Window width                   [W x]

Memory used: 15291   Memory free: 17351   Window size:    64
Word count:   2537   Line count:    337   Current doc: LETTER
----+----+---------+---------+---------+---------+---------+---
Command ?
```

Each of these commands will be discussed in detail in Part II. It is from this menu that you load text files from the disk, save your documents to disk, establish the global find and change strings, check various information on the status of your document. Press <command> again, and you will be returned to the Text Entry/Edit screen. The Command Key serves as a switch to take you from the Text Entry/Edit screen to the Command Menu and back again. (It also serves as an Abort Key to stop what you are doing and return to the Text Entry/Edit screen.)

While at the Command Menu, type the following, being careful to include the space where indicated:

L GOTHIC

Press <enter>, and the sample document entitled "Gothic Suspense, Chapter 15" will be copied from the disk and put into the buffer, and you will be taken back to the Text Entry/Edit screen. The sample document is now on the screen. But what happened to what you had typed before going to the Command Menu? Like Clementine, it is lost and gone forever! We did this intentionally to show you that if you want to save your work, you must save it before loading something else!

If you know your document is going to be more than a page long, make it a practice to save it early, and update it as you are working. For example, when beginning this part of the manual, the author entered the header lines and the title of the document, then saved it to disk. Although only 13 lines in length, with six lines containing only the <enter> symbol, this created a new file on the disk, and provided insurance against the loss of work through momentary power outages (it will happen). Any time your attention is taken away from what you are doing, simply press <command>-S-<enter>, and the work you have done will be saved to the disk.

CAUTION

Pressing <command>-S-<enter> while you are making these changes will result in permanent changes unless you have write-protected your disk! If you have more than one drive available and your system disk is write-protected, you will then make a copy of the "improved" letter on another drive. If this is the case, you may load either one by calling for the document with the proper drive specification. If you have a tape system, and update your document as you work, make certain you are saving to a clean place on the tape. If you overwrite a document, you will never be able to recover it.

If you wait to save your document until you have finished, or until you reach a "natural breaking place," you are asking for trouble. (If you claim your attention is never diverted from your work, it is obvious you never have to answer a telephone, a doorbell, get up and stretch your legs while you refill your coffee cup, discuss anything with co-workers. But certainly, you must get tired of the chattering of the monkeys and the screaming of parrots on that desert island!)

Learn how to move around a document. Use the Arrow Keys to move in the four directions indicated. Use the Page Up Key to move your cursor up a full screen, and Page Down Key to move it down a screen. The Left Line Key will postion the cursor at the beginning of the current line, and the Tab Key will move the cursor to the next tab stop. Two other movement controls are very useful. <funct>-T will take you to the first line of the document (Top), and <funct>-E to the end.

If you are accustomed to "spacing over" a line with a typewriter, you're in for a surprise if you try it with MSCRIPT, which thinks you want to overtype what's there with spaces. Go ahead, try it. The gaps might look interesting, but they're really not very practical. Try overtyping, perhaps your name and address, perhaps the Gettysburg Address. What a mess!

Remember that we said when you took a file from the disk, you only put a copy of it in your buffer? You haven't made an illegible mess of your sample letter at all. Simply press <command>, then reload the document by typing L GOTHIC, and you'll have another copy of the document to experiment with.

There is another sample document on the disk called LETTER, and you may want to have a look at it. Return to the Command Menu, and notice the R option. "Restart MSCRIPT" is how you clear the current document from the memory to start a new one. Press <R> now, and return to a clean screen. Load the document called LETTER and look at it.

This document is ready to go to the printer. In addition to the text lines, you see strange lines that begin with ">". These are Format Lines, and are for the purpose of giving directions to the printer. They do not show in the final copy after being printed. We'll explain each of these lines and all the commands later. For now, ignore them. Just make some changes in the letter itself for practice.

After some practice with GOTHIC and LETTER, restart MSCRIPT once again, and enter some text of your own, but don't worry about format lines. Just enter some text, practice moving around

in it, overtype some of it, and just get the feel of moving the cursor. Once you're doing that without much conscious effort, you may go to Part II and learn how to insert and delete text.

Backing Up Your Work

We cannot emphasize too strongly the importance of making backups of your work. Even the best quality disks are fragile and susceptible to being damaged. The most common dangers are usually found right on your desk where you work. Because disks are magnetic storage devices, anything that creates a magnetic field can scramble the information on a disk. The little magnetic memo holders you use to clip notes to your file cabinet are strong enough to erase part of the disk; the power supply in a stereo or radio could do it; the electromagnet that drives the hammer to ring your telephone can do it. Get ALL such items away from your disks. Never use pencils or ball point pens to write on disk labels; use only felt-tip markers. A better practice is to write the label before attaching it to the disk.

While floppy disks are flexible, they can be permanently damaged by bending. Insert them into the drive doors carefully, and never touch the shiny surface. A fingerprint can cause the loss of days or even weeks of work.

The owner's manual that came with your computer tells you how to make a backup of a data disk, and recommends that you do this at the end of each day. In some situations it is better to back up your work more often; this depends on the importance of the work, and only you can be the judge in that matter. The few minutes spent making a backup copy is time well invested, as is the cost of the extra disks. If you feel that the cost of extra disks is exorbitant, consider how much time you spent creating those documents. All the text for this manual is contained on one 80-track, double density data disk, which cost about $3.00. Would you be willing to retype this manual for $3.00? Disks are cheap; time is not.

Part II
MSCRIPT In Use

Unlike some other word processors, MSCRIPT uses only one screen for entering text and editing what has already been entered. There is no need to switch back and forth from one function to the other. To demonstrate MSCRIPT'S ease of editing, type in the following paragraph. Don't be concerned if you make a few mistakes; if you don't have any mistakes, you won't have anything to edit!

> Most people who use a word processor for the first time are awed by the power that is suddenly available to them. There is no need to fear this new tool; MSCRIPT is easy to learn and easy to use. If you take one thing at a time and understand what it does before going to the next, you'll find yourself at home with MSCRIPT and able to use all its features in a few hours. It is not necessary to attend classes for days to learn MSCRIPT, nor do you need to spend 12 to 18 hours listening to tapes and copying dozens of complicated examples.

Now take a good look at your text. If you've made any errors you correct them in one of three ways:

1) Overtyping
2) Deleting Characters
3) Inserting Characters

Before going any further, look at the Help Menu. Press the Function Key, and the letter H. You now see something similar to the menu on the next page:

Editor commands - Press FUNCTION+LETTER

B - Block begin or end marker	C - Copy block within text
D - Delete char under cursor	E - Position to end of text
F - Repeat last FIND command	G - Imbed printer control code
H - Display this help menu	I - Open insert block
M - Merge text blocks	N - Force new page
P - Select print menu	Q - Quit changes to line
R - Remove marked block	S - Enter sub-delete mode
T - Position to top of text	U - Unmark block in text

MSCRIPT word processing software
----+----+---------+---------+---------+---------+---------+---
Copyright 1983, Micro-Systems Software, Inc. Key to continue

Each of the letters, when pressed in conjunction with the Function Key while you are working at the Text Entry/Edit Screen, will perform the action indicated. These are powerful editing commands, and we'll work with them in a logical sequence. Pressing any key while you are at this menu will return you to the Text Entry/Edit screen. For now let's look at the three most commonly used functions in editing a document. These are Overtyping, and use of the I, D, M, S, and Q commands. Press any key and return to the Text Entry/Edit screen. Now, take a good look at your document.

1) Overtyping

If you have typed "MSCRITP" instead of "MSCRIPT" simply overtype your corrections. In practice, you'll find that when you are entering text and make a mistake, you'll backspace with the left arrow and overtype the error. Text entry is actually overtyping a line of spaces, and MSCRIPT, as you remember, recognizes spaces as valid characters.

2) Deleting Characters

If you find you have typed an extra letter in a word, place the cursor over the letter, then press <funct>-D, and watch the letter disappear. When the letter was deleted, all the remaining text shifted one space to the left. If you continue pressing <funct>-D, you'll see a stream of characters disappear. Again, they are like Clementine, lost and gone forever, so don't hold down any longer than you need to. (There are ways to delete more than a character at a time, and these will be discussed later. For most of your editing, though, you will find that this is how you'll delete a letter or a word or two.) Notice that if you delete several characters and there is room on the line for the first word on the next line, it will move up and adjust itself

(or, "wrap around"). If you delete any characters in the first word on a line and it will fit on the previous line, it, too, will move up ("wrap") and adjust the rest of the text, but will wait until you move the cursor off the line.

Look at these examples:

> when the modern office practices make
> the handling of words much easier than
> before.

If you delete the word "when" from the first line, MSCRIPT will adjust the text by wrapping "the" up to that line:

> modern office practices make the
> handling of words much easier than
> before.

You never need to make a decision about what to do with gaps in the text when you delete any characters from your document. MSCRIPT does this automatically, and requires no action on your part.

3) Inserting Characters

If, however, you have typed "MSCRPT" you will need to insert the I between the R and the P. To begin inserting, place your cursor on the character where the insertion is to be made, in this case over the P. Press <funct>-I, and MSCRIPT opens your text from that position to the end of the line, while moving the remaining text down to the next line. You now type the missing letter, and have "MSCRI" followed by empty spaces. To close this gap, press <funct>-M, and you'll see the text snap back into place, with the corrected word.

At this point, look at the Help Menu again. You may have wondered about the terms "Open insert block" after the I, and "Merge text blocks" after the M. You remember that a block was defined as any amount of text. When you pressed <funct>-I you actually instructed MSCRIPT to take your text (which itself is a block), and create three new blocks. The first block is the text from the beginning to the cursor. The blank spaces from the cursor to the end of the line are the "insert block," into which you inserted the letter I. The remainder of your text, which was moved down to the next line, is the third block. When you pressed <funct>-M, you instructed MSCRIPT to merge the three blocks of text into one. The only limitation on the amount of text you may insert is the available memory in your buffer. If you attempt to insert more characters than you have free in your

buffer, you will be stopped with an error message on the bottom of the screen, saying, "Out of memory!" and will not be able to enter more. To return to your text, you'll have to press the Command Key. The text you tried to insert will be gone, but you will not freeze up the machine and lose your text, as happens with other word processors.

In addition to closing up your text with the <funct>-M, you may close it simply by moving the cursor off the line. Note that when you do this, a space is inserted after the last character you inserted. MSCRIPT facilitates the insertion of a word by automatically adding a space at the end of the word. Try the following example, by typing in these two lines:

```
This is a sentnce.
This not a sentence.
```

In the first case, you need to insert a letter, and in the second, a word. Place your cursor over the second "n" in "sentnce" and press <funct>-I. You now have:

```
This is a sent
nce.
This not a sentence.
```

Type the "e", then move the cursor off the line, and you have this:

```
This is a sente nce.
This not a sentence.
```

However, if you had followed the "e" with <funct>-M, the extra space would not be there and would not have to be deleted.

Now insert the word "is" into the second sentence. Place your cursor over the "n" in "not" and press <funct>-I. Type the word "is" and move the cursor off the line, and you have inserted the word and the necessary space. Had you closed the text with <funct>-M it would then be necessary to go back and insert a space.

Deleting More than One Character

In addition to deleting characters one at a time, MSCRIPT allows you to delete an entire word, delete from the cursor to the end of the video line, or to remove all spaces to the beginning of the next visible character. By pressing <funct>-S, you will see something similar to this displayed at the left of

the status line:

[W]ord [H]ack [R]emove [SPACE BAR]

To delete from the cursor position to the next space, press W; the word has been deleted with one keystroke. This feature is very handy when you want to change a word. Simply delete the old word, then insert the new one.

"Hack" sounds as though you are going to chop something to pieces, and this is exactly what the term means. If you choose this option from the sub-delete menu, pressing H will delete from the cursor to the end of the current video line. Please note that it is the video line, not a sentence. This command allows you great flexibility in rearranging or editing certain kinds of text, such as tables and columnar material. This feature will be explained in greater detail in Part III (Special Applications).

If your text contained the lines

whenever another copy of the document
is needed for any reason, it can be done
easily.

and you wanted to delete the words "of the document," place the cursor on the letter "o" of the word "of" and then <funct>-S; pressing <H> will "hack" off the rest of that video line, producing this:

whenever another copy
is needed for any
reason, it can be done easily.

Now, simply by moving your cursor to the line above or the line below, you will have this:

whenever another copy is needed for
any reason, it can be done easily.

The final option from the sub-delete menu will remove a long row of spaces on a line, from the cursor to the next character. This is an extremely powerful feature when working with tabular text, and like the Hack command will be illustrated in detail in Part 4. In addition, it provides another method of deleting several words, which some people find easier than repeating the <funct>-D, or using the [W] command several times. Simply "wipe out" the offending words by overtyping them with spaces, position the cursor at the beginning of the string of spaces and use the [SPACE BAR] to close the gap.

An example of this is seen in the following:

```
typewriters, duplicating machines, file
cabinets, duplicating machines, file
folders, and all the other items that
```

If you were to overtype the redundant words with spaces, you would have this:

```
typewriters, duplicating machines, file
cabinets,                          file
folders, and all the other items that
```

Now, move your cursor to the left, leaving one space after the comma at the end of "cabinets" and press <funct>-S, then <spacebar>, and you will have this:

```
typewriters, duplicating machines, file
cabinets, file
folders, and all the other items that
```

Notice that the words on the next line did not wrap up to fill the line. This was not an error on MSCRIPT's part. Have you ever wished you could go back and undo something you didn't really want to do? Perhaps all you really wanted to do was delete one word, not hack off the rest of the line. The <funct>-Q command allows you to change your mind if you delete something that should't have been deleted, with certain limitations. Notice that when you delete a character, a word, or several words, that MSCRIPT does not close up the text and fill the line with text from the following line until you move the cursor off the line. In the event you find you have made a mistake (deleting the wrong word or character, or too many words, or perhaps your mind was wandering and you were hacking instead of thinking), if you press <funct>-Q, the line will be restored to its original condition before you started deleting things.

CAUTION!! If you move the cursor off the line, the deletions will be permanent (remember Clementine?), and using <funct>-Q will have no effect. Once your text has been reformatted (closed up), what you have deleted is lost and gone forever. While recovery of the original line is possible until you move the cursor to another line, do not form the habit of sloppy and irresponsible deletions! Think about what you are doing.

Block Manipulation

One of the most powerful features of a word processor is its ability to manipulate a "block" of text. A block of text, you will remember, is any amount of text that begins and ends with a "block marker." The text within the markers may be as short as a single character, or as long as the entire document. MSCRIPT allows you to define a block of text, then copy it to other locations within the document, move it, or delete it. You may have more than one block marked; there is no limit on the number of marked blocks in your text. The only thing you must remember is that when you give MSCRIPT a command to copy or remove a block, it will search for the first block, and perform the desired action on that block. In Part 4 you'll learn how this can be used to advantage. Generally, though, you will be working with one block at a time.

Look at the Help Menu again. You have four commands that deal with blocks. <funct>-B inserts a "block marker" in your text at the cursor position. <funct>-C will make a copy of that block anywhere you desire. <funct>-R will delete the block, and <funct>-U removes the block markers. Let's see how these work.

Marking A Block

To mark the beginning or ending of a block, simply place your cursor at the desired location and press <funct>-B. A graphic symbol, called a "block marker," is inserted at this point. It is not necessary that you mark the beginning first; you may mark the end of the block, and then the beginning, if that suits your purposes better. When you command MSCRIPT to do something with the block, it will find the first block marker in your text and understand that this is the beginning of your block. It then searches for the next block marker, and knows that this is the end of the block.

Copying A Block

Once you have inserted the block markers, you have a "marked block" of text within your document. Remember Miss Pelham? She was asked to move a paragraph from page four and put it in page one. Most writers do extensive "cutting and pasting" when rewriting, and often dread doing another revision because of the need to retype everything when it just needs rearranging. MSCRIPT eliminates the need to retype anything. You need only to mark the paragraph with block markers, put your cursor in the correct position on page one and press <funct>-C, which will insert a copy of the marked block at that place. If you want to make a copy of it in several locations within your document, you may copy it as often as you desire by pressing <funct>-C.

Deleting A Block

While it is possible to make many copies of a marked block, actual practice shows that you will most often simply want to move it to another location. When you have completed copying the block, press <funct>-R to delete it from its original position. This is known as moving the block. Four simple commands are all it takes. Mark beginning, mark end, copy, remove the block. In Part 4 we'll see a few different ways to use marked blocks.

The <funct>-R command is very handy when you want to remove a large portion of your document from the beginning or the end. Suppose you want to print only a short portion of a long document, and it is somewhere in the middle of the body of text. Locate the beginning, press <funct>-B, press <funct>-T to return to the top of the document, <funct>-B to insert the other marker. Now mark the end of your desired text with <funct>-B, press <funct>-E and get to the end of the document, mark it with <funct>-B, then press <funct>-R twice, and you have only the short piece of text you want to print.

This feature is very useful when you want to create another document using the same tab settings, window size, etc. After you have saved the first document, simply mark the entire thing as a block, delete it by pressing <funct>-R, and you will have a "clean slate" for the next one, while preserving all the tabs, etc.

CAUTION!

Because the <funct>-R command removes the first block you have marked, you must be careful when working with more than one block. Also, the <funct>-U will unmark all blocks. This is a very important reason to remember what you are doing. Working with a single block at a time can eliminate much grief.

Global Find and Change

Remember Miss Pelham's dilemma? Was that word "Procedure" or "proceedure"? This is one of about a dozen common words that have been giving her trouble ever since she was in the fourth grade. If she had a dollar for every time she's had to turn to her dictionary and check the spelling, she could take six months off and do the Grand Tour in style. Having some commonly used words that simply don't register correctly does not mean a person is less than intelligent. In many cases it is a result of experiencing shame or ridicule from a teacher or classmate when learning to spell the word for the first time. Every time the

word is encountered, the subconscious mind says, "Wait a minute; we have a problem." Even if the text is examined by a proofreader program (a special program that reads your document and points out misspelled words), when the incorrect word is displayed on the screen as incorrect, it looks correct to the operator, and use of a printed dictionary must be resorted to. (Most versions of MSCRIPT can be used with any good proofreader, such as Electric Webster. Check with your dealer.)

By using the <funct>-F command to find the misspelled word, and <command>-C to change all the ocurrances of error to the correct spelling, long documents may be edited quickly to correct these errors.

If you have a long document, say a chapter of a book, and need to locate a particular word, name, or phrase, the global find will take the cursor to it immediately. This can be a real time saver when editing documents that are many pages long.

Another use of the global find and change function is in the case of a word or name that is clumsy or difficult to type. Miss Pelham found the name "Pelczarski" awkward, and because it would occur a number of times, typed "ppp" in its place. When she was finished, she used the find and change function to change every instance of "ppp" to "Pelczarski."

Finding A String in Text

To use this feature, you first press <command> to get to the Command Menu. Notice that the F and C command examples shown in the right hand column have no space following the letter ([Fstring], [Cstring]). The term "string" indicates the combination of characters you are searching for. Look at these three examples:

```
Fword
F word
F wor?
```

If you enter the command "Fword", when you press <enter> MSCRIPT will find the first occurance of the letters "word" after the cursor. The search begins at the cursor postion, not at the top of the document, unless your cursor is already there. It will find "word", "words", "Foreword", "key-word" or anything else having the string "word" in it.

If you enter "F word", MSCRIPT treats the space as a valid character, and will ignore "Foreword" and "key-word" in its serach.

Notice the question mark in "F wor?". This is known as a "wild card" and will find any occurrance of " wor" followed by any other character. Thus MSCRIPT will find " word", " work", " world", " worm", " worrywart" and " worthless" in its search. This feature can be very handy in searching for combinations of words. If you are not certain about the position of the letters "c" and "z" in the name "Pelczarski", you could ask it to find "Pel??arski". If you were to ask for "Pel??" it would find not only "Pelczarski" but "Pelham" as well.

When you enter the Find String (which can be as long as 30 characters) at the Command Menu and press <enter>, MSCRIPT returns to the Text Entry/Edit screen, and places the cursor on the first occurrance of the string. To find the next, press <funct>-F, and MSCRIPT will place the cursor on the next on. This will continue each time you press <funct>-F. If it does not find the string you desire, or if it has found the last one, you will see the message "String not found!" on the Status Line. This tells you that the string either was not in the document, or that you have found every one of them.

Changing the String in Text

In order to change "ppp" to "Pelczarski" you must use the related command "Cstring". Remember that the same rule applies here in regards to spaces within the string as in the Find String. You must have entered a Find String before you can use the Change String. Once you type "Cstring" at the Command Menu and press <enter>, all occurrances of the Find String will be replaced by the Change String. You must be aware of two things when using the Change String, however.

After typing in the Find String and pressing <enter>, you will then be returned to the text, at the first occurrance of the Find String. While at the Command Menu, if you press <enter> after typing "C" and a Change String, you are telling MSCRIPT to change all the Find Strings to the Change String, from the cursor to the end of your text. If you do not want all of them changed, you must change them individually. If you have not specified a Change String, but simply typed "C", all of the Find Strings will be deleted.

Again, this should present no problem as long as you are aware of what you are doing.

Appending Documents

One of the powerful features of MSCRIPT is its ability to create new documents from several existing documents. Think of this as "daisy chaining" one to another. Any document on any disk can be appended to the one in the buffer, i.e. added to the end of the document in the buffer. Suppose you had a number of short "boiler plate" documents (ones used over and over to fill in space) on your disk, and wanted to send a letter to someone using four of the 25 or so standard pieces of "boiler plate" you have. In the sample document "LETTER" the descriptions of the three models of Widgets was boiler plate. When the typist got to that point in the letter, she simply went to the Command Menu, told MSCRIPT to add that particular item, then returned to the letter to finish typing after the Widget descriptions. Once something is typed and corrected, it never has to be retyped! You can even decide to make boiler plate of a portion of a document that was created months before, by editing out all of the document except the part you want for boiler plate, then saving the remainder under a new name.

The Command Menu shows this as:

A - Append doc. to buffer [A doc.name]

At the Command Menu, typing A followed by a space and the name of another document (whether boiler plate or a full document), then <enter> will add the requested document to the end of the text in your buffer. The only limit on how many things you can put together in this fashion is the amount of free memory in the buffer.

This operation is sometimes called "daisy chaining," because it works like building a chain of daisies to make a long necklace. An example of how this can be used to cut down typing time is seen in the memo handed to Miss Pelham:

"Send Pay-or-Else letter to John Doe
17, 18, 20, 24 and 33."

Miss Pelham addresses the letter to John Doe, then appends four documents she has called BP17, BP18, BP20, BP24, and BP33. By appending these five pieces of boiler plate to the name and address, she has saved herself three and a half pages of typing something that she had already typed and corrected. Countless hours of repetitive work can be saved in this manner.

Setting Tabs

When you first start MSCRIPT, the tabs are set at 5, 10, 20, 30, 40, 50, and every ten spaces to the maximum width of the page. Their locations are shown by the plus signs in the Tab Line at the foot of the Text Entry/Edit Screen. Frequently you will want to change these settings, substituting locations of your own. From the Command Menu, type T followed by the numeric values of the settings separated by commas. For example, if you type:

T 5,12,28,45,52,58<enter>

you will be returned to the Text Entry/Edit screen, and pressing the Tab Key will move you to those locations on the line. You need to be aware that MSCRIPT considers these settings from the beginning of the printed line, not from the left edge of the paper. They may be changed at any time while you are working on your document, affording unlimited flexibility.

A word of caution, though. When you use the command <R> to restart MSCRIPT from the Command Menu, the tab settings will revert to their default condition, i.e., as they were when you first started MSCRIPT. Therefore, if you want to preserve any settings for another document, after you have saved it to the disk use the block commands to delete the document. If you have frequent need for particular settings, you will be shown an easy way to recall them later when we discuss Format Lines.

Adjusting the Width of the Window

When you start MSCRIPT, the size of your Window is the width of your Video Line, and usually 2 lines less than the maximum displayed on your monitor. These lines are the Tab Line and the Status Line at the bottom of your screen. The length of your video line is the same as the width of your screen. The video line length has nothing to do with the length of a printed line, which is set either from the Print Menu or in a Format Line. There are several reasons for wanting to change the size of your window. Many people find it easier to read a short line while they are entering text; they have vision problems that are aided by narrower lines of print on the screen. At the other end, there will be times when it will be desirable or even necessary to have video lines longer than your screen width. If you are creating a document that will be printed on 15" wide paper, containing 132 columns, and need to see exactly how each line will look when it is printed, you will need to adjust the Window size. If you are creating columnar documents that are wider than

your screen width, you can do it only by adjusting the Window to match the width of the printed material.

At the Command Menu, simply type <W> and then the width of the video line you desire, up to a maximum of 230. You will be returned to the Text Entry/Edit Screen and the video line length is now the value you entered after the <W>. When your cursor reaches the end of the line, the line will move off the screen to the left until you enter a word that wraps around to the next line. At this point, the words in the window shift automatically; MSCRIPT does this with no further action on your part.

Saving Your Documents

After you have completed entering the text for your document, save it to disk. If you have not yet saved it, or if it is a new document, you must give it a name. Press <command>-S, then type a space and name the document, then press <enter>. MSCRIPT will then make a copy of the document in your buffer and write it to the disk.

If you are unfamiliar with valid names for your document, you must check your computer manual. DOS's (disk operating systems) vary in the number of characters permitted, as well as the kinds of characters. All permit an extension of the name, and this extension can be used to advantage in organizing and identifying what documents are stored on the disk. Think about how you file paper documents in the old fashioned office. Each drawer of a file cabinet normally contains one type of document, or a variety of documents pertaining to one person, one company, or one type of activity. Inside each drawer are a large number of file folders. Each file folder has an identifying label on it, or a color coded label, which helps in keeping things organized and easy to locate.

If you think of each data disk as a file drawer, you can think of each document on the disk as a file folder. The name of the document, with its extension, tells you what the document is. For example, if you keep all correspondence with the Ajax Corp. on one disk, you could name your documents something like "JAN15", "FEB4/BAL" (or "FEB4.BAL" depending on your machine), to indicate letters written on January 15 and February 4. The "BAL" could be your code for "demand for BALance due." We mentioned above using "BP17" for Boiler Plate item 17. What labels you use and what extensions you put on them are strictly your choice, but keep them meaningful and most important, keep them unambiguous. Be careful that you do not use the same name twice when saving a document to your disk. If you do, MSCRIPT

will consider the most recent one to be an update of the one already on file, and the previous one will be overwritten and lost. If you are not certain if a document name already exists, read the directory first. See the following for instructions on how to read your directory.

DOS Commands from MSCRIPT

If your Command Menu does not have the following line, skip this section:

DOS commands/exit [D command]

Depending on your machine, you have a number of DOS commands at your fingertips while you are working with MSCRIPT. The most frequent command will be to read the Directory, to see what documents are already on the disk in order copy one to your buffer for editing or updating. You will also need to know what is already on the disk to prevent duplicating a document name, thus losing the previous document with that name. Check the pages in this manual that pertain to your particular computer to see which DOS commands you have available.

Printing Your Document

The purpose of a word processor is to print words on paper. So far we have discussed many of the operations of MSCRIPT, but have not yet actually printed a document. If you create a document and intend to save it, make certain you have saved it before doing any extensive editing, and save it after editing and before printing. MSCRIPT will not damage the text in your buffer during the printing process, but saving before printing is a form of insurance against one of Murphy's Laws, which states that "Power failures will occur only if your text has not been saved to disk."

Press <funct>-P and examine the Print Menu, which looks similar to the one on the following page:

```
MSCRIPT Print Menu

JU - Justification      N       SS - Single Sheet        N
PL - Page length       60       PS - Page spacing        6
LM - Left Margin        0       LL - Line length        60
LS - Line spacing       1       PN - Page number         1

Type the two-letter command and new value to change the
setting (example: JU=N will turn justification off and
PN=15 will set the starting page number to 15).

Press <enter> to print entire document.
Enter <PF> to print from cursor to the end of document.
Press <command> to return to Text Entry/Edit screen.
----+----+---------+---------+---------+---------+---------+---
Command ?
```

The default values (the built-in values which you see when you go to the Print Menu) may or may not be what you need for your particular document. If you just press <enter> MSCRIPT will interpret these values as meaning that you are using continuous paper (roll or fan-fold), 11 inches long; it will not be justified; you want 60 lines printed on each page; the lines will be 60 characters long; there will be a 6 line break between pages (½ inch margin at each the top and bottom); you want your document single spaced, and begin numbering the pages with 1; no spaces will be provided for a margin on the left side of the page.

You may change any of these values by typing the appropriate two-letter command followed by the new value you desire. In many instances no changes will be necessary. If you are wondering why MSCRIPT sets its defaults to 0 for the left margin, it is because there are many printers available that set margins by a switch setting on the printer itself (hardware control), rather than from the word processor program (software control).

The Format Line

In addition to setting the printing format from the Print Menu, you may do this with Format Lines in the document itself. This is necessary if you want to change any of the settings within the body of the document. All of the commands below may be entered into your document at the beginning or later. Many of the Format Line commands are identical to the Print Menu commands. With one exception, they are followed by Y, N, or a value, which is represented here by xx:

```
>JU=Y      (Begin justifying text)
>JU=N      (Stop justifying text)
>CE=Y      (Begin centering text)
>CE=N      (Stop centering text)
>LM=xx     (Set left margin to xx)
>LL=xx     (Set line length to xx characters)
>LS=xx     (Set line spacing)
>FR=Y      (Begin printing flush right)
>FR=N      (End printing flush right, return to normal)
>PN=xx     (Begin page numbering with xx)
>*         (Non-printing remark line)
>HI=Y      (Begin hanging indent)
>HI=N      (Return to normal indent)
>#x=x/x    (Define special print code)
```

In using these Format Line commands, you must remember that each must appear at the beginning of a line, immediately following an <enter> symbol, and that they are terminated by another <enter>. Several of them may be combined on one line, separated by a space or a comma, and they may be either upper or lower case. In all cases, the Y or N represents a switch to turn it on or turn it off. (There are eight additional commands, all of them dealing with Headers and Footers, and we'll look at those later.)

The SS, PL, and PS commands are seldom used in a Format Line; most often they are set at the Print Menu.

At this point, load the document named LETTER and let's examine the Format Lines. The next page shows you what it looks like on the screen, and the page after that is what it looks like when printed.

```
>LM=10,JU=N,LL=60,¶
>CE=Y¶
©+Williams Widget Works©+¶
2727 West Warrentown Way¶
West Warrentown, Wyoming  65432¶
December 12, 1983¶
>CE=N¶
¶
¶
Mr. Ernesto Pinto, Mgr.¶
Consolidated Distributors¶
Waverly, Florida  33877¶
¶
¶
Dear Mr. Pinto:¶
¶
    It was a pleasure meeting you at the Rocky Mountain Widget
Workshop last week. Because of the great interest in our
display by over four thousand visitors, it was impossible for
us to answer all your questions at that time. The Workshop was
the most heavily attended trade show I have had the pleasure of
attending, and we were caught short on the samples we
distributed to our visitors.¶
¶
>LM=15,LL=50,JU=Y¶
Under separate cover I'm sending you samples of our three most
popular widgets; a standard ©-2" Blue Widget©-, a ©-3" Red
Improved Widget©-, and our newest model, the ©-4.5" Super
Heavy Duty Green Giant Widget©-, which has been successfully
implemented in thousands of end user applications.¶
¶
>LM=10,LL=60,JU=N¶
    Delivery on the Red and Blue models is from stock; the
Green Giant is currently running 30-45 days behind, and until
we open our new manufacturing facilities here in Wyoming, the
delivery time will continue to be slow. If you have any
questions on our Widgets or other products that are not
answered by the brochures, please don't hesitate to call me.¶
¶
    I'm looking forward to a great future for all of us, Mr.
Pinto.¶
¶
                                    Sincerely,¶
¶
¶
¶
                                    Geoff "Dutch" Masters¶
                                    V.P. Sales¶
```

Williams Widget Works
2727 West Warrentown Way
West Warrentown, Wyoming 65432
December 12, 1983

Mr. Ernesto Pinto, Mgr.
Consolidated Distributors
Waverly, Florida 33877

Dear Mr. Pinto:

It was a pleasure meeting you at the Rocky Mountain Widget Workshop last week. Because of the great interest in our display by over four thousand visitors, it was impossible for us to answer all your questions at that time. The Workshop was the most heavily attended trade show I have had the pleasure of attending, and we were caught short on the samples we distributed to our visitors.

> Under separate cover I'm sending you samples of our three most popular widgets; a standard 2" Blue Widget, a 3" Red Improved Widget, and our newest model, the 4.5" Super Heavy Duty Green Giant Widget, which has been successfully implemented in thousands of end user applications.

Delivery on the Red and Blue models is from stock; the Green Giant is currently running 30-45 days behind, and until we open our new manufacturing facilities here in Wyoming, the delivery time will continue to be slow. If you have any questions on our Widgets or other products that are not answered by the brochures, please don't hesitate to call me.

I'm looking forward to a great future for all of us, Mr. Pinto.

Sincerely,

Geoff "Dutch" Masters
V.P. Sales

The LM, LL, JU, and CE Commands

The first two lines of LETTER are Format Lines. Note that each of the commands could have been written on a separate line, provided it begins with ">" and is terminated with <enter>. The ">" must be the first character on the line; not even a space may appear before it. (Spaces are valid characters, you remember.)

```
>LM=10,JU=N,LL=60¶
>CE=Y¶
```

The "¶" symbol appears where <enter> was pressed, and may or may not be what you see on your screen, depending on your machine. The commands on the first line tell MSCRIPT you want your left margin to begin 10 spaces from the edge of the paper, do not justify the text, and use a line length of 60 characters. The second line tells MSCRIPT to begin centering each line. After the date line, the next command is to stop centering the text. The next two lines show that no text has been entered, but <enter> was pressed.

The next Format Line is this:

```
>LM=15,LL=50,JU=Y¶
```

This tells MSCRIPT to change the left margin to 15, shorten the line to 50 characters, and justify the text. This type of "indented block" stands out visually because it is different from the balance of the letter. At the end of the paragraph, the next Format Line is encountered:

```
>LM=10,LL=60,JU=N¶
```

MSCRIPT now returns to the previously defined format. Again, each of these commands could have been put on a separate line, and the order in which the commands appear on a line makes no difference.

With this document (LETTER) in your buffer, it is time to go to the Print Menu again to print this letter. If you are using standard letter size paper (8½ x 11 inches) ignore what you see on the menu, which shows that your left margin is set at 0. Any values in a Format Line in your text will "override" values on the Print Menu.

Get your printer ready by making sure it has paper, ribbon, is connected to your computer, and the "on-line" and power switches are actually on. Position your paper so the print head is about ½ inch below the top of your paper. The first line of

text will be printed wherever you position the print head. If you want to provide extra space for a top margin, you do this by using linefeeds (pressing <enter> on a blank line) above the text. Now, press <enter> and MSCRIPT begins printing the letter. You will see that none of the Format Lines is printed; they exist only to give commands to the printer.

When the document has finished printing, go back to the Print Menu and see what's happened to the value for the left margin. Whatever values were last encountered by MSCRIPT have been transferred to the Print Menu.

Underlines and Bold Face

In the sample letter, the name of the company was printed in bold face, and the three widget descriptions were underlined. Look at the document on your screen. Immediately before the bold face printing was started, a graphic symbol appears, followed by a "+", and the same thing appears at the end. Before and after the underlining, the same symbol appears, followed by a "-". These are the commands to turn on the bold face or underline, and to turn them off. If you refer to the Help Menu, you'll see the G command, which is used to tell MSCRIPT that what follows is a printer command that does not need to be on a Format Line; these commands appear within the body of the text itself, or "imbedded" in the text. MSCRIPT has defined the two most widely used print enhancements, bold face and underlining, with the "+" and "-" characters. You may type these in as you create your document, or add them later. If you add them after creating the document, it is not necessary to use the <funct>-I to insert them. Position your cursor over the first letter to be underlined, press <funct>-G, and the symbol will be inserted, as well as a space after it. You then type the "+" or "-" for whichever you wish.

Note that the commands for bold face and underlining work like toggle switches. The first one turns on the feature, and it will remain on until MSCRIPT encounters the next one. If you want to underline centered headings like this:

This Is An Underlined Title
Made Easy by MSCRIPT

you need to start and stop the underlining on each line. This is not necessary within the body of text, but in titles this is what will happen if you do not turn it off at the end of each line:

This Underlined Title
Is a Disaster!

Usually, you will underline only a few words in your document, or a title, so it is not often you will encounter this situation. But remember that for titles you must turn it on and off on each line.

The FR Command

There are three basic formats in printing non-columnar text. It can be (1) Justified, (2) set Flush Left (also known as Ragged Right, because of its uneven right margin), or (3) Flush Right. This paragraph has been set using the FR, or Flush Right format command. Normally, text is set flush left or justified, but there are occasions for setting it flush right. Advertising copy is frequently placed on the left side of an illustration, and the even right hand margin makes a pleasing visual appearance. This paragraph has also been set with lines that are shorter than the rest of the manual.

When using the Flush Right format, it is not necessary to indent at the beginning of a paragraph, since the left margin will be ragged anyway. Also note that you cannot have Justified text and Flush Right active at the same time.

If you want justified text to appear as a block at the right side of the page, you must reset the left margin, turn off the Flush Right Format with "FR=N" and turn on the Justification with "JU=Y." This is how this paragraph was formatted.

The HI Command

MSCRIPT provides you with another Format Line command that is a desirable feature of a word processor. This is the Hanging Indent, sometimes called Reverse Indent. It gets its name from the fact that the first line of text is printed at the left margin, while all other lines (justified or unjustified) are indented, just the reverse of the normal paragraph indent. Thus

the term "Reverse Indent." "Hanging Indent" is the proper term as used in the printing industry. A Hanging Indent is frequently used in places where you must catch the eye of the reader with a term, a word, or some phrase for quick and easy identification. The Glossary at the end of the manual is set with a hanging indent. The command is:

>HI=xx

The "xx" is any numerical value that is shorter than the Line Length. If you want to indent all lines (except the first line) 8 spaces, use a Format Line that reads, ">HI=8" and the result is this:

Hanging Indents are frequently used in glossaries, catalogs, price lists, and footnotes. With many other word processors, you must terminate each line and indent all lines after the first by tabbing. It is a very difficult task, if not impossible, to justify text that way.

MSCRIPT makes hanging indents simple. Just type your text as you would any other, paying no attention to line length or word wrap. After you press <enter>, MSCRIPT sets the first line to the left margin you have specified, and indents the balance automatically, even justifying it if you desire.

This is an extreme example, but illustrates what happens when you change the indent value to 20 with the format line, "HI=20." MSCRIPT will format your text in any fashion you desire.

The * Command

It is frequently desirable to have non-printing comments in your documents. If you send form letters to clients who are overdue on their accounts, it would be embarrassing to send a demanding letter that should have been preceeded by a phone call, for example. For documents that are updated frequently, price lists, or catalogs, for example, you might want to show the date it was last updated. If you send different types of letters on different styles of letterheads, a comment line will remind you which letterhead to use with which document. The Comment Line is a Format Line using the following command:

```
>* This line will not print.
```

When MSCRIPT encounters the ">" it knows it is dealing with a printer command, and knows it does not print anything after the ">", but looks for a two letter format command. In this case, when it finds the "*" it recognizes it as being a Comment Line, and ignores everything else until after the next <enter>.

If you have a document that makes use of tab settings other than the default settings, and want to be able to set them quickly again for another document without having to check the location of each one, use a Comment Line at the beginning. This is also helpful if you need to remember to reset the width of the window:

```
>* Tabs 12,26,39,48,72 - Window 80
```

When you load this document the Comment line is a reminder that you need to set tabs and window before continuing.

The >#x Command

Most printers are capable of printing special characters that do not appear on the keyboard. The "¶" is an example, as are à, ½, ©, and £. Because each printer has its own set of special characters, you must refer to your printer manual to learn what they are. It is essential that you know whether it is a printable character or non-printable character. For example, if your printer is capable of overstriking one character with another by backspacing, the character that causes the printer to backspace is not a printable character. If it has the capability of printing superscripts or subscripts, the commands that instruct it to roll a half line up or down are non-printable characters as well.

MSCRIPT allows you to define up to 10 special printer codes at a time. This is done in a Format Line:

>#0=xx/x,#2=xx/x,#3=xx/x, etc.

You remember that to underline or boldface your text, you pressed <funct>-G, then typed a "-" or "+" in the space after the symbol that appeared. This same operation with the <funct>-G is used to tell your printer to use one of the special codes you have defined in a Format Line. In the case of the "¶" this was defined as follows:

>#0=175/1

The ">" tells MSCRIPT that we are giving it a printer command. The "#0=" tells it that we are defining a code that will follow <funct>-G as the number "0" (it could be any number from 0 through 9). "175" is the decimal value of the character "¶" on the particular printer we are using for this manual, although it is probably different on your printer. (Your printer might not have a "¶" at all.) The "/1" tells MSCRIPT that the code "175" produces a printable character. Non-printable characters will be followed by "/0" and MSCRIPT will ignore them when justifying text.

Most printer manuals list the special characters on a reference page, and show Decimal, Hexadecimal and sometimes Octal codes. Use the Decimal Value when you define any special printer codes, otherwise unpredictable gibberish could result, and might resemble the I.R.S. Handbook.

In addition to the two codes built into MSCRIPT (underline and bold face), you have another ten at your command at any given time. You are not limited to ten, however, since if you find you need more you may redefine them at any time with another Format Line.

MSCRIPT's use of special print codes allows you to do many things that are impossible with many other word processors. The following line was possible because of the way these codes were defined:

T̸h̸i̸s̸ l̸i̸n̸e̸ h̸a̸s̸ b̸e̸e̸n̸ d̸e̸l̸e̸t̸e̸d̸/

In this case a combination of backspaces and slashes were used to overstrike the characters in the line. Because the <funct>-G inserts two characters on your screen, if you intend to use any print enhancements in your text, it is best to add them after you have created the document. The line above, while

less than a printed line in length, occupies two video lines on the screen, because of the imbedded codes. This is especially important to remember when you are working with columnar text, since each imbedded code will move your text two spaces to the right. This is a case of "what you see is not what you get."

Headers and Footers

Most documents of more than a page in length will need some method of identifying each page as being a part of the entire document. Frequently this will be nothing more than a page number, although it could also identify the document by title, date, or anything else desired. This information is usually placed in a Header or a Footer. A Header will appear at the top of a page, and a Footer at the bottom of a page. (This manual contains only Headers.)

MSCRIPT allows you total flexibility in the placement of headers and footers. If you think for just a moment about the placement of a page number, there are six obvious locations on any page: Upper left, upper right, lower left, lower right, centered at top, or centered at bottom. You must be able to specify where to place it. Let's suppose you want to place the title and page number at the top of each page. If you use a header with the title on the left of the line and the page number on the right, it will appear that way on all pages. Of course, in the case of a document that is printed on one side of the page only, this is fine. If you want to print on both sides of the sheet, it is customary to print the page numbers for odd pages on the right side, and for even pages on the left. This necessitates creating two headers (or footers), one for odd pages and one for even pages.

The TT, ET, and OT Commands

If you want to use the same header on all pages, you will use the format command ">TT=/.../.../..." This tells MSCRIPT that what follows is a Top Title, and to print it on every page of the document. The slashes indicate the position on the line for the title. What you put after the "=/" and before the second slash will appear flush left; between the next two slashes will be centered, and after the third will be flush right. This alone gives you tremendous flexibility. For example, the following format line will print the words, "Annual Widget Report" flush left, and "December, 1983" flush right:

```
>TT=Annual Widget Report//December, 1983
```

There is no need to be concerned about how many spaces to

put between the items on the left and right; MSCRIPT does all the figuring for you.

If, however, you want different headers on odd and even pages, you would create two header lines, one using the ">ET=/.../.../..." format command, the other ">OT=/.../.../..." for Even page Top title, and Odd page Top title. The following format lines will create two different headers, the first appearing on even numbered pages, the second on odd numbered pages:

```
>ET=/Annual Widget Report//December, 1983
>OT=/December, 1983//Annual Widget Report
```

Until now, nothing has been said about how to number your pages. Using the symbol "$" anywhere in a header (or footer) will tell MSCRIPT that you want your pages numbered, and to place the number where the "$" is found. The next examples show how to number even and odd pages using the same text, but placing the numbers on the outer corners of the printed pages:

```
>ET=Page $//Widget User's Manual
>OT=/Widget User's Manual//Page $
```

Of course, if you simply want to center the page number, and dress it up a little, you could use this:

```
>TT=//-{ $ }-/
```

The BT, EB, and OB Commands

Footers work the same as headers, with the difference that the B in these commands replaces the previous T, and the line will be placed at the bottom of the page. MSCRIPT's versatility thus allows you to define as many as two headers and two footers for your document. The following format lines will place the headers on even and odd pages respectively, and center the page number on all pages:

```
>ET=Section 7B//Widget Maintenance
>OT=Widget Maintenance//Section 7B
>BT=//-[ $ ]-
```

(Notice in these examples for the headers, no slash was used after the "=". When you want something to print flush left in a header or footer, the first slash is optional, but the others are not. Notice the footer command, which has two slashes. This will center the page number. If you want something to print flush right, with nothing else printed in the other positions,

the command would be ">TT=///xxx.")

There will be times when you desire to turn off a header or footer, perhaps because of the inclusion of illustrations. This is done quite simply by inserting a format line without any defined text:

```
>TT=
>BT=
```

Headers and footers will then stop printing. If you want to turn them on again, simply issue a new command with another format line.

The HM and FM Commands

Headers and footers occupy one line each, and are calculated in the total number of lines to be printed on each page. Because they appear to be the same as any other line of text to MSCRIPT, you must specify how many lines to insert between these lines and the body of the text. This is done by using the Header Margin and Footer Margin commands. If you want three blank lines between your header and the body of the text, and two lines between the body and the footer, these format lines give the commands:

```
>HM=3
>FM=2
```

MSCRIPT will calculate the total number of lines in headers, footers and their margins (seven, in this case), and include the seven lines when calculating the number of lines to print before going to the next page.

If you turn off a header for any reason, you will find that the text will begin printing at the top of the page, where the header had been. In the case of footers, it will print down to the line where the footer had been. To prevent this, you can tell MSCRIPT to "print" blank lines in their places:

```
>TT=/
>BT=/
```

This will work with any of the title commands (TT, ET, OT, BT, EB, OB).

The margins used by headers and footers are the ones that were defined and in use just before the headers and footers were defined. If you move your left margin in five spaces, and

shorten the line length 10 spaces within your document, these changes will have no effect on the headers and footers. This is a feature of MSCRIPT that is not found in many other word processors, and prevents unwanted changes to your headers and footers if you change margins for a portion of your document that begins on one page and is carried to the next. The text margins will be narrower (in this case) than the headers and footers, preserving the appearance of the final document.

The PN Command

One of the commands at the Print Menu is PN, which defines the starting Page Number. This may also be defined in a format line with the following command:

```
>PN=xx
```

In the case of long documents, such as book manuscripts, the page number is usually defined in a format line at the beginning of each chapter. The format lines at the beginning of a chapter may read something like this:

```
>LS=1,LM=10,LL=64,PL=56,PS=10,PN=187¶
>TT=Gothic Suspense - Hess//- $ -¶
>HM=3¶
¶
¶
>CE=Y¶
Chapter 15¶
>CE=N¶
```

MSCRIPT will begin numbering Chapter 15 with page 187. By now, all the commands you see should be familiar to you. There is only one thing left before we go to the next part of the manual.

Forcing the End of A Page

When MSCRIPT begins printing your document, it heeds the advice of the Walrus when he told Alice, "Start at the beginning, go until you come to the end, and then stop." It will pay attention to all the format comands you have given it, print and special characters you have defined, print to the margin and line length you have specified, and print the correct number of lines on a page, from the beginning until the end. Usually this is good, but there are times when it could be disastrous as far as your document is concerned. If you tell MSCRIPT you want 58 lines printed on a page, it will print 58 lines, then skip to

the top of the next page and print another 58 lines. This can ruin something like the following:

```
####################################
##                                ##
##            NOTICE              ##
##              of                ##
##            ANNUAL              ##
##            PICNIC              ##
##                                ##
##   All personnel in Dept. 3     ##
##   and Warehouse 27 will be     ##
##   in attendance at the 1983    ##
##   Widget Picnic to be held     ##
##   in Warrentown Civic Park     ##
##       December 31, 1983        ##
##     4:00 p.m. until 1984!      ##
##                                ##
##            (byob)              ##
####################################
```

If the line reading, "Widget Picnic to be held" happens to be the 58th line on the page, the box will be chopped off and continued on the next page. It sometimes happens that the title to a section is left on one page, separated from the text it belongs with. To prevent this, you may issue a "New Page" command with <funct>-N. Insert this at any place in the text, and a graphic symbol will be inserted and the text separated the same as if you had pressed <enter>. When MSCRIPT finds this command it stops printing, counts the necessary lines to the end of the page, and begins the next page.

If you have need of several blank pages where you wish to paste in illustrations or other art work, you may use the <funct>-N as many times as you need empty pages. All headers and footers will appear in their proper places, and page numbering will remain correct.

Part III

Some Special Applications

In earlier sections of this manual I said I'd show you some special applications that are easily handled by MSCRIPT and which can save you many hours of work over a period of time.

The examples given in this section are among many I have devised myself, and some I've learned from others. As you work with MSCRIPT and find other ways to cut down the hours you spend "reinventing the wheel," think about passing them along to us, and we'll share them with all MSCRIPT owners through the newsletter we publish, the DOSPLUS News Information Center. The first is one I've been using myself for several years, thanks to a friend who had a problem in organizing things.

The Random Note Finder

My friend has been a society photographer and reporter for many years, and wanted to know "if my computer could help me keep track of all my negatives." My first reaction was to suggest a data base management program, but the second reaction followed close on its heels. The photographer/writer in question is a lady in her seventies, who modestly describes herself as "the most disorganized person in the county." I agreed to investigate the problem for her, and found she had been keeping her negatives in manila file folders for more than ten years, with no more information on the folders than the month and year the photographs had been taken! If she needed to locate a negative, she had to refer to her day-to-day appointment books, then hunt through the file folders.

Most data base management programs are fussy about the information that is entered. The date goes in a certain place, and provision is made for particular information. My friend is not one who can be compelled to "color within the lines," and any data base management program I'd ever seen would have been too restrictive in her case. While I was studying the problem her phone rang, and moments later she said, "That's the kind of thing that happens all the time! That was Mrs. Blank, and she wants a copy of that picture I took of her three years ago with Burt Reynolds at the Breakers Hotel. Or was it five years ago at the DAR luncheon at The Collonades? She can't remember."

The pieces began to drop into place, and before the evening was over, she was started on what has proven to be the greatest time-saver of her life. She simply began going through her negative files and using her word processor to store information like this:

> Sep 23, 1980. Urban League Charity Ball. Palm Beach Hilton. Jennifer Blank, Sam Goldman, Henrietta Withington.
>
> September 10, 1976. Republican Women's Club Holiday Inn PGA Blvd, Harold Swartz, Henrietta Withington, Myrtle Dothentwine.
>
> Martha Swartz, Jennifer Blank, Downtown Development Agency, Howard Johnson's -- May 6, 1979
>
> Myrtle Dothentwine presented the Republican Women's Club Annual Appreciation Trophy to Sam Goldman (3rd from left) at a luncheon last Thursday at the Hyatt Hotel. -- June 9 1982

Unlike a data base management program, her word processor doesn't care about the order in which she enters the information, nor does it care if she fills in blanks for two names or twenty; she can use "Sep," "Sept.," or "September" for the date. As soon as her system was instituted, she began typing her captions on the word processor, and saving them all to her "Random Note File," as she called it. Now when Mrs. Blank wants a copy of a photograph, she uses a global find command to locate the negative. If she knows Mrs. Blank wants a copy of the photo that includes Burt Reynolds, it's a simple matter of searching a note file on disks that are organized with six months worth of information on a disk. Every time she finds "Jennifer Blank" she can scan the other names and find the one that also has Burt Reynolds.

This simple system can keep track of a postage stamp collection, or magazine articles you want to locate. There is no need to make multiple entries as with file cards; the cross references are all in the single entry and can be located with a global find. It is also used as a recipe file by many housewives. Writers can use it to keep track of all the notes they scribble on matchbook covers, cocktail napkins, business cards, scraps of paper and other things because they don't have a notebook handy. While working on an article or a book, the notes can be appended to the document, particular details located, then deleted when ready to return to work.

The Letterhead and Envelope

The sample letter on your MSCRIPT disk (LETTER) includes the name and address of the Williams Widget Works, and is used to produce correspondence on plain paper. There is another file called LETTER/HD which contains only the company name, address and date. A few blank lines are included, and the words "Name here" indicate where to begin typing the addressee's name and address. The document is loaded, the date overtyped and then the cursor is moved to the last line and the letter is begun. The number of keystrokes saved in typing the format lines and header is about 150; if it is used only three times a week, that amounts to 23,400 keystrokes a year, or about 6,000 words. The body of the letter itself in LETTER contains 216 words. By using a file that contains the heading, enough time can be saved to produce more than 25 letters.

If you send out more than three letters a week the chances are that you are using printed stationery. Having a document that contains the format lines, enough blank lines to position the date under the printed letterhead, and then the lines to the name and address block will save a considerable amount of time. And if you write to one person or firm with any regularity, it pays to keep a file with that information ready to use.

Another document on your disk is called "ENVELOPE" and shows how simple it is to address an envelope. Again, if you are writing to one firm regularly, keep an envelope file on the disk with that correspondence. Having a number of short but useful files saves more time than the cost of a few disks.

Mailing Lists

Everybody keeps lists of names and addresses. We're not talking about business mailing lists of thousands or even hundreds of names. If you are maintaining that type of mail list, you probably have a mail list program that is capable of sorting all the married women over the age of 35 who have purchased a minimum of $1000 worth of widgets within the last 90 days, and who live in zip codes 33401 through 33409.

The lists we're dealing with here are the short ones, such as members of a civic club, church groups, scout troops, and so forth. Even if you do have a mail list program, you probably don't have all your personal lists stored that way. You can easily create a file that will print labels, if you'd like. Just create a document that looks like this:

```
>PL=5,PS=1
>* Widget Works Beer Drinking Team
John Smith
123 Fourth St.
Anytown, Florida  33409
>* Feb 84
>* (305) 555-1212
>* President
>* Wife Sally
<funct>-N
Bill Jones
5300 Park Lane, Apt. 34-B
Anytown, Florida  33406
>* Aug 84
>* (305) 121-2555
>* Sgt. at Arms
<funct>-N
Vladimir Pelczarski
8862 Forbes Road
Anytown, Florida  33405
>* Dec 85
>* Wife Roberta
>* Current State Guzzling Champion
<funct>-N
```

Notice that you have three lines containing the actual information that will appear on a mailing label, and as many non-printing comment lines as you like for additional information. At the end of all the lines for each name, you use a New Page function, to tell MSCRIPT to go to the next page to print the next line. Since the Format Line has defined the "page" as having five printed lines and one line to space to the next "page" MSCRIPT will print the three lines for name and address, encounter the <funct>-N and skip to the top of the next "page." In this case the page is one inch from top to bottom, since standard self-adhesive labels are 15/16 inch, or a full inch. Why five lines on the page instead of three? You could, if you want, print a label showing the month club dues expire, or any of the other information, simply by deleting the ">* " at the beginning of the line. To print a reference sheet with all the information, remove the ">* " at the beginning of each line. If you have a number of them to remove, use a global find and delete ("F>* ", "C"<enter>). Because you are not changing any of the information on the disk file, these one-time changes will not wipe out any work you have done previously.

This is how the list will print:

John Smith
123 Fourth St.
Anytown, Florida 33409

Bill Jones
5300 Park Lane, Apt. 34-B
Anytown, Florida 33406

Vladimir Pelczarski
8862 Forbes Road
Anytown, Florida 33405

Form Letters

The same list you use to create labels can be used to generate personalized form letters. Load the list into memory, and mark each name and address as a block. Then create the letter, or if it is already in a file, append it to the mailing list. Position your cursor where the name and address are to appear, and press <funct>-C, which will copy the first block. Press <funct>-R and remove the block from the list, then print the letter. When you return to the Text Entry/Edit screen delete the three lines and simply repeat the process for each letter.

Other Uses for the Mailing List

In addition to printing your return address on an envelope, you can print as many of one kind of label as you like. Most of us use labels to identify books, boxes, cartons, etc. Homemakers who take dishes to church dinners need to have a way to put their names on the items they take. Self-adhesive labels stating:

```
**************************
*   From the Kitchen of  *
*       Kamelle Hess     *
**************************
```

help give credit for a great dish of shrimp creole, as well as let everyone know who owns the empty dish.

There is no need to return to the Print Menu each time another of these single labels is desired. Enter the Format Line, the lines to appear on the label, then mark the label as a block and copy it several times. If you want a large number of labels, after making half a dozen copies, delete the end marker and put another one at the end of the sixth one. Now when you copy, you'll be adding six at a time, or however many you have in the block.

Self-adhesive labels are available for freezer use, and having a supply of them on hand to identify what's under the wrapping and the date it was stored can help organize your food supply and prevent loss from spoilage because of age.

Contents:_____
Date:_____
Use before:_____

Columnar Text

Straight columnar text presents no problems. Simply set your tabs at the required locations, and use the Tab Key to move to the next area. If you remember, we said we'd show you some ways to use the Sub-Delete commands. One again, press <funct>-S and review them before we show you some applications. Let's assume you are setting up a list simply to show names, ages and length of membership in an organization. If you haven't planned it carefully, this is what can happen:

Name	Age	Yrs. Mbshp
Adams, George	54	16
Baker, Charley	44	8
Smith, George	3	23
Pelczarski, Vladimir		

Our old friend Pelczarski has a name longer than you had counted on, and has run into the column of figures for the age. No need to panic, and because this is not a typewriter, no need to retype anything. Simply put your cursor into the blank area between Name and Age, and insert two spaces. Then reset your tab stops, and you're ready to continue.

Another problem is leaving too much space to have the results pleasing to the eye:

Name	Age	Yrs. Mbshp
Adams, George	54	16
Baker, Charley	44	8
Smith, George	3	23
Pelczarski, Vladimir	26	5

This is easy to correct, and shows another of the Sub-Delete commands. Put your cursor a space or two after "Vladimir," press <funct>-S, press <spacebar>, and the line will snap into place, again leaving the cursor in postion. Move the cursor up and down into the other lines and repeat the process. This could also be done by deleting the same number of spaces in each line with <funct>-D, but deleting the spaces with a single keystroke is much faster.

Yes, Miss Pelham? Oh, you say that George Smith can't possibly have been a member for 23 years if he's only three years old? Good point, and an excellent way to show you how to use another of the Sub-Delete commands. You could simply overtype the figures, but you might find it easier to place the cursor after "George," press <funct>-S, press H and hack off the line. Then you can tab across the line and insert the correct figures.

If you remember, we said that print enhacements should be added to the document just before printing it, and that this kind of columnar word made it essential. Suppose you wanted to underline the names of the officers on that list, like this:

Name	Age	Yrs. Mbshp
Adams, George	54	16
Baker, Charley	44	8
Smith, George	23	3
Pelczarski, Vladimir	26	5

What it looks like on your screen is this:

```
          ©-Name©-                    ©-Age©-     ©-Yrs.
Mbshp©-¶
          Adams, George               54          16¶
          ©-Baker, Charley©-              44          8¶
          Smith, George               23          3¶
          ©-Pelczarski, Vladimir©-    26          5¶
```

Hardly the kind of thing you'll enjoy editing. That's why we suggest that you add all the print enhancements after you've done all other editing and just before you're ready to print.

Programs and Bulletins

Church bulletins and banquet programs present another type of columnar material which is easily handled by MSCRIPT. In this case you have text which is both flush right and flush left, with leaders connecting the two sides:

```
**********************************************************
*          *            Program             *            *
* ------------------------------------------------------ *
* Call to Order.....................Allan Toor, DTM *
* Invocation.....................Rabbi Jacob Herschel *
* Pledge to Flag.....................George Adams *
* Welcome.....................Mayor Alice Futz *
* Musical Interlude.....................Tommy Tucker, DG *
*    Supper                                              *
* Main Address.....................Geoff "Dutch" Masters *
* Awards.....................Vladimir Pelczarski, ATM *
* Final Remarks.....................Allan Toor, DTM *
**********************************************************
```

An easy way to handle this type of material is to determine first the total width and enter the row of asterisks for the top border. After each item is entered on the left side, fill in the line with periods. The asterisk in the right hand side of the heading was used as a guide to the length of the row of periods. Immediately following the periods the names were typed, along with the space and asterisk for the right border. After pressing <enter>, the cursor was moved into the line of periods, and enough periods were deleted to align the borders.

```
    ********************************************************¶
    *          *            Program             *          *¶
    * ---------------------------------------------------- *¶
    * Call to Order.....................Allan Toor, DTM *¶
    * Invocation..................Rabbi Jacob Herschel *¶
    * Pledge to Flag.........................George Adams *¶
    * Welcome.................................Mayor Alice Futz *
¶
    * Musical Interlude.....................Tommy Tucker, DG *¶
    *    Supper                                            *¶
    * Main Address.........................Geoff "Dutch"
Masters *¶
    * Awards.............................Vladimir Pelczarski,
ATM *¶
    * Final Remarks.........................Allan Toor, DTM *¶
    ********************************************************¶
```

A little imagination in setting up this kind of text will produce programs that are attractive, and a pleasure to do because it has eliminated the old-fashioned counting of spaces.

Backup Copies Are Insurance

Although we covered this before, it is so important it must be restated. One of the first pieces of advice you learn from your computer's owner's manual is to make frequent backups of your work. Some people argue that it costs too much to have two copies of everything on disks. If you consider that even one of the best quality disks available costs less than an hour of work at minimum wage, I defy you to show me how you're going to save money if you destroy even part of the information on a disk!

Disks at best are fragile. The most common causes of losing data are found on or near your desk. The most common is magnetism. Do you have any of those cute little magnetic memo holders that you put up on the file cabinets? Dropping one on a disk can erase the information on it, and make it impossible to recover. What makes the bell ring in your phone? Only a small electromagnet that drives the hammer. Do you keep a radio on the desk to provide soothing music while you work? Any radio or TV has transformers that are powerful electromagnets, as well as other litle goodies that emit RF waves.

When you buy disks each one is packaged in a paper envelope that usually contains some advertising from the manufacturer. It

also contains a list of Don'ts for the care and handling of the disk. Heed these warnings well, for it's very frustrating to lose a disk full of files simply through mishandling. In the long run, you, the operator, are the greatest cause of disk failure. Buy good quality disks, for the same reason you buy good quality shoes. While a $60 pair of shoes sounds expensive, they are a better buy than 10 pairs at $6. The same is true of disks. Use a good quality disk that costs a bit more than the El Cheapo, but will last much longer and be much more reliable.

There are two ways to make backups of your work. One is to heed the general advice to make a backup of the disk at the end of the day, or at the end of the session. The other method is simpler, and provides yet another bit of insurance, as we'll show you.

If your system permits you to swap disks in mid-session, have two formatted disks to receive your document when you save it, not one. Get into the habit of saving it, then swapping data disks and saving it again on the other disk. In the long run, it takes less time, for you are making two copies of the current document, and not having to wait while you make a copy of EVERY document on the disk. If you have three or more drives, save it twice, once on each drive with a data disk. The added insurance is preventing you from making a backup of the empty, formatted disk over the disk containing the documents. It has happened to more than one person! Simply specifying the wrong drives is all it takes.

Again, get in the habit of saving your work at frequent intervals. The few seconds it takes to update the current document and do it a second time on another disk is far less at the end of a week than having to redo anything too large to print on a 3x5 file card!

Full Screen Editor for Programmers

If you do any BASIC programming, you can write your programs with MSCRIPT. They will be saved as ASCII files, which take up a bit more space on your disk than compressed BASIC, but you have the advantage of being able to work with a full screen editor rather than a line editor. The only function you won't have available is automatic line numbering, but features like the global Find and Change will far outweigh the loss. You may renumber lines if you wish, or even develop a program and debug it on the screen, then change PRINT statements to LPRINT with ease, and know that you haven't missed any of them.

* * *

There is no way we could list all the ways you'll be putting MSCRIPT to work, nor can we anticipate all the ways you'll find to make it easier to use and more versatile. That's why we'd like to hear from you with your ideas, and we'll share them with all owners in the DOSPLUS News Information Center, the newsletter that goes out to all registered owners of Micro-Systems Software applications packages or DOS's.

Glossary of Word Processing Terms

abort = To stop or cancel a procedure or operation in progress. In MSCRIPT this is accomplished by pressing the <command> key.

access = The ability to obtain data from memory, or place it into memory.

applications software = Programs that perform specific types of input and output to achieve a useful result. Word processing, real estate management, and inventory control are examples of applications programs. Compare applications software with systems software.

ASCII = Acronym for American Standard Code for Information Interchange, pronounced "asskey." A set of symbols and controls that can be printed by most printers, and which are found on ASCII-type keyboards.

audio-cassette storage = Standard home cassette tape recorder used as a storage device for a word processor in the absence of disk storage. Tones are recorded and played back representing digital 1's and 0's.

auxiliary storage = Mass storage. An alternative type of storage, usually employing equipment or media that is slower but less expensive.

backup = (1) Duplication of a document or group of files on disk or tape onto a separate storage medium to be preserved in case of loss or damage of the original. (2) An exact copy of a document or group of files.

basic input/output system (BIOS) = That portion of the CP/M operating system which cosists of drivers and other software that manage the peripheral devices of the computer.

baud rate = A measure of the rate of speed at which data is sent from the computer to the printer, or other device. The number of bits transmitted per

second (bps). Common baud rates are 110, 300, 1200, and 9,600. When divided by ten, it gives an approximate number of characters transmitted per second.

bidirectional printer = A printer which prints characters in both directions as the print head travels across the paper. Bidirectional printers are usually faster than unidirectional printers, which must return to the left of the page before printing the next line.

BIOS = Basic Input/Output system.

bit = Binary digit. The smallest piece of information recognized by a digital computer. Represented by a 1 or a 0, signifying "on" or "off," "yes" or "no." Groups of bits (usually eight) are called bytes. One byte represents one character.

block = Any defined amount of text that can be acted upon as a whole. A block may be as short as a single character, or as long as an entire document. The text is defined as a block by the insertion of block markers at the beginning and end. Once so marked, a block may be copied, moved or deleted.

boiler plate = Previously prepared documents that are used as fillers for other documents.

boot = The process of intializing a computer and preparing it for use. This usually includes clearing everything from memory and loading the "bootstrap" program, the few instructions needed by the computer to start itself. See your computer manual for the proper method of "booting up" your system. Some computers have different procedures for "cold boots" and "warm boots."

buffer = An area of memory reserved for creation and editing of text; a temporary holding area. In this MSCRIPT manual, "memory" and "buffer" are used interchangeably, for the most part.

bug = A defect in a program which results in incorrect results or a crash.

byte = A group of eight bits which comprise one character. Each of the 128 ASCII characters or commands may be expressed with an eight-bit byte.

carriage return = A non-printable character that causes the cursor of a CRT or print head of a printer to return to the beginning of the line. It is activated by pressing the "Carriage Return" key, which is sometimes known as the "Enter" key. It is the terminator used to indicate the end of a set of instructions, for example, a Format Line. In this manual it is indicated by the symbol <enter>.

catalog = The directory of files, or documents, in storage on the disk.

cathode ray tube = Also referred to as CRT. A monitor on which is displayed the text of the document. With some systems, it is a standard black and white or color TV set.

character = Any one of a set of letters, numbers, symbols or unprintable commands recognizable to the computer. Each character occupies one byte of memory.

character pitch = A designation of the width occupied by a printed character on the paper. The pitch tells how many characters may be printed in the space of one inch of the text line. Most frequently used pitches are 10, 12 and 15.

character printer = A device which transfers single characters to the paper. The most popular character printers are dot-matrix, daisy wheel, and ink-jet.

code = (1) The system of characters used to represent data. (2) The instruction set comprising a computer program. (3) A special instruction imbedded in the text which does not print but controls some special printer function.

cold start = Booting the system after a crash or when it has just been turned on.

command = An instruction given to the computer telling it what to do.

command key The key which switches from the Text Entry/Edit screen to the Command Menu, and back to the Text Entry/Edit screen. It is also used to abort any operation or command during execution.

command menu = In MSCRIPT, a list of choices available to the operator.

compatible software = (1) Software that is able to be run on different computers without modification. (2) Software that shares features with another package, enabling one to use information created by the other. Electric Webster and MSCRIPT are compatible.

computer system = The machine and all its peripheral components which, when instructed by appropriate software, functions together to perform useful tasks.

control character = An unprinted character that causes the computer system to perform a function, such as turning on the printer. Some control characters are used in conjunction with other characters to define unique commands.

control key = A key that controls some program function or cursor movement. When pressed simultaneously with another key special functions are performed. In MSCRIPT the Function Key acts as a control key, and is referred to in this manual as <funct>.

cpi = Characters per inch. See character pitch.

CP/M, CP/M-80, CP/M-86 = Control Program for Microcomputers, an operating system developed and copyrighted by Digital Research, Inc.

CPU = Central Processing Unit. The heart of the computer, or the brain of the computer. CPU's in common use are the Z-80, 8080, and the 8086/8088.

crash = (1) A malfunction of hardware or software resulting in los of data or physical damage. (2) In a hard disk, the action of the head coming into unwanted contact with the platter, resulting in physical damage to the head and/or platter.

CRT = See cathode ray tube.

cursor = A special character displayed on the Text Entry/Edit screen indicating the exact position at which a character will be entered or deleted.

cursor control keys = Keys which move the cursor around

the document without overtyping. In MSCRIPT these are the four arrow keys, the Page Up Key, Page Down Key, Left Line Key, and Tab Key.

daisy wheel = A printer element shaped like a wheel made up of radiating spokes, with embossed characters on the ends. So-called because the wheels resemble daisies. The wheels for any printer are interchangeable, providing a wide variety of type styles and faces.

data = Information that is input to a computer and then processed so that it may ultimately be output in meaningful form.

dedicated word processor = A computer system that functions only as a word processor, as opposed to one that is capable of running a variety of software to perform different functions. MSCRIPT is a word processor that is designed to function on a wide variety of non-dedicated computer systems.

default = A value supplied by MSCRIPT in the absense of values input by the operator.

descender = The portion of a printed or displayed letter that extends below the base line. The letters g, j, p, q, and y have descenders. Some dot-matrix printers are incapable of printing descenders.

directory = The list of files in storage. The directory itself is a file.

disk = See floppy disk.

disk drive = A random access storage device which is able to read and write to magnetic disks.

diskette = See floppy disk.

disk operating system (DOS) = A series of programs that control space allocation, and keep track of all files written to magnetic disks. It also controls the saving and reading of files, and performs all the tasks that control the operation of the computer. DOSPLUS, CP/M, MS-DOS and TRSDOS are examples of disk operating systems.

disk pack = See hard disk.

disk system = The total of the components required for disk storage, including the computer, disk controller, disk drives, disks and software.

display = The output device used to observe the text visually without printing it on paper. This is usually the CRT or TV set.

docname = See filename.

documentation = The instructions that are supplied with software and/or hardware that explains how it is used, or how it operates. This manual is the documentation for MSCRIPT.

DOS = See disk operating system.

dot-matrix = A pattern of dots. Dot-matrix printers form characters on the paper by firing individual wires against the paper, usually in a 5x7 or 7x9 pattern. Greater numbers of dots can be printed, creating better resolution and clarity by some printers.

electrostatic printer = A non-impact printer that forms the characters by creating an electrostatic charge on the paper in the form of the characters, then dusts the paper with powdered ink, and finally melting the ink with heat to form a bond with the paper.

embedded command = See imbedded command.

end of page marker = An invisible code that stops the printer at the end of a printed page to allow the operator to insert another sheet of paper. In MSCRIPT this provided by the program if the operator enters SS=Y at the Print Menu.

enter = Used interchangeably with Carriage Return. In lower case ("enter") it means to press ENTER after typing a command. In this manual, it is designated by <enter>.

extension = See file name.

external memory = Same as mass storage.

fan-fold paper = Continuous sheets of paper containing

perforations, folded in an accordion fold. Used in a printer to eliminate the need for the operator feeding a new sheet of paper for each page.

file = A group of data assembled for one particular purpose and considered a unit. Each document saved on magnetic storage is a file. In addition to documents, files can be system operating files, directory, etc.

file name = The title given to the file to identify it as being unique. On the Command Menu as throughout MSCRIPT it is called "docname," meaning "Document name." Most systems allow file names of up to eight characters, although some permit more. All permit extensions of three letters, which further aid in identifiction. The extension is separated from the file name by a period or a slash, depending on the requirements of the disk operating system. A group of files relating to each other could share a common name or extension, for example. GLOSSARY/PT1, GLOSARY/PT2, GLOSSARY/FIN are three files for this glossary of terms. JONES.ENV, JONES.HDR, JONES.INV are three files for a client named Jones.

firmware = Programs or data contained in ROM. These cannot be changed by software, and are not lost when power is interrupted.

floppy disk = A circular sheet of mylar coated with a magnetic medium for recording and storing data. Several sizes are in common use, and the correct terminology differentiates the sizes and contrasts with "disk," which usually refers to a hard disk. A diskette is 8" in diameter; a minidisk is 5.25" in diameter; a microdisk is 3" in diameter. All, however, are commonly referred to as "disks." The floppy disk is sealed in an envelope, a protective square cover lined with a soft material that protects it from scratching as it turns within the envelope. The envelope contains various openings: the oval slot, or window, for the read/write head(s); the index hole; the centering hole, and the alignment slots.

font = The style of characters on a print wheel or controlled by software or firmware in a dot-matrix printer.

format = (1) The physical appearance, or layout, of the document when printed; the specific margins, headers, and placement of copy on the page. (2) To prepare a blank disk to receive data. On some systems this is known as initialization.

freeze = A condition caused by an error in the program, faulty hardware, or incorrect commands given by the operator; the computer recognizes no input from the keyboard, and the only remedy is to reboot the system.

friction feed = A method used to transport paper through a printer, comprising rollers that grip the paper as they turn, as in a standard typewriter.

full-page display = A terminal that displays at least 80 columns and 55 lines of copy, which is the average for an 8½ x 11 inch page. The CRT is mounted vertically in such terminals.

function code = Non-printing codes that generally are used to control a peripheral device such as a printer. These control line feeds, printing of non-standard characters, etc.

Function Key = In MSCRIPT, the designated key to be used in conjuction with another to perform a designated function. In this manual, use of the Function Key is designated <funct>.

garbage = Meaningless and unwanted data displayed or printed. A single, incorrect code sent to the printer may result in an otherwise perfect document being printed as garbage.

global find and change = The function that enables the operator to search for every occurance of a particular series of characters in a document and change them to some other series of characters.

half-height drives = A disk drive that is approximately one half the physical thickness of a standard drive. Sometimes referred to as Slimline Drives.

hang-up = See freeze.

hard copy = A permanent copy of a document, usually on paper, as opposed to the temporary display on the screen.

hard disk system = A storage system that utilizes rigid platters as opposed to floppy disks. The platters are arranged in stacks which may or may not be removable from the system. The platters are organized so that all tracks in vertical alignment with each other are considered a "cylinder" and are considered one continuous track; this aids in keeping search time to a minimum. Hard disk systems are able to store much more information than floppy disk systems, and are more reliable. The are also considerably more expensive.

hard-sector disk = A floppy disk on which the sectors are designated by individual sector holes.

hardware = The physical components of a word processing system; the computer, printer, disk drives, etc., as opposed to the programs, which are software. A piece of hardware containing programs or instructions for the computer, such as a ROM chip, are known as firmware.

head = The electromagnetic device that records data onto the disk and reads data from the disk.

head crash = Unwanted physical contact of the read/write head with the surface of the platter, resulting in damage to either or both.

Help Screen = A fuction available which displays information for the assistance of the operator without the necessity of referring to the manual. In MSCRIPT this is obtained by pressing <funct>-H.

horizontal scrolling = The ability of MSCRIPT to shift the window horizontally to view more characters than are able to fit the width of the screen.

imbedded command = Characters or instructions inserted into the text that do not print, but give commands to the printer, such as change the line length, change the line spacing, etc.

impact printer = A printer which transfers ink (or carbon) to the surface of the paper by striking through a ribbon.

initialization = (1) Preparing a disk for use by organizing the surface into tracks and sectors. (2)

Setting all starting values needed prior to creating or printing a document. In MSCRIPT, <command>-R initializes the program.

ink-jet printer = A printer which produces characters on paper by spraying droplets of ink under electrostatic control. Such printers are capable of extremely high resolution, color reproduction, and quiet operation.

interface = The hardware and/or software needed to allow one device to work with another. The power cord which allows your computer to interface with the power company is properly called the AC Power Interface.

I/O = Input/Output, referring to transfer of data from one part of the system to another, primarily the disk drives and the printer. The computer operating at a billionth of a second is said to be "I/O bound" by a printer that operates at a hundredth of a second, necessitating a loss of time because the computer sits idle until the printer is ready for the next instruction.

justification = Spacing of text to produce evenly aligned right and left margins. This is usually accomplished by inserting extra spaces between words. This glossary is justified, for example, whereas most of the balance of the manual is not.

K, KB = Kilo, KiloByte; approximately a thousand (1024) bytes, or characters.

keyboard = The typewriter-like device that accepts the input of the operator and sends instructions to the computer. Various keyboards are found with word processors: ASCII, Dvorak, QWERTY, etc.

laser printer = A printer that focuses laser beams for forming images on photosensitive drums, similar to the xerographic process.

letter-quality printer = A printer employing a daisy wheel, thimble, "golf ball" element, or other type of fully-formed character to strike the ribbon and transfer the letter to paper.

library = A collection of programs or documents that are used frequently. Form letters, standard

headings, boiler plate and envelope forms are examples of what can be termed a library.

line feed = The mechanism in the printer that causes the paper to advance by one line; the control, code or character that activates the mechanism or causes the cursor to move down one line on the screen.

line height = The height of the printed line, measured in number of lines per inch.

line printer = A printer that produces a full line of printed characters at a single stroke, as contrasted with a character printer. Line printers are extremely fast, operating at speeds up to 3,000 or more lines per minute.

machine = The common term used in the industry for computer.

machine dependent = Capable of being used only on the particular computer for which is was designed or written. While MSCRIPT is available for a wide variety of machines, each version is machine dependent.

memory = A temporary storage area for programs such as MSCRIPT, or data, such as a document. Main memory, or internal memory, is that which is immediately accessible for use, and includes RAM and ROM. The term as used in this memory excludes any form of external memory, which is used only for storage.

menu = A list of options available to the operator which is generated by the program and displayed on the screen. MSCRIPT contains a Help Menu, a Command Menu, and a Print Menu.

microcomputer system = A complete system which includes the CPU (central processing unit), input/output interfaces and devices (keyboard, cassette tape, disk drives), power supply, CRT display (or TV), and memory. Generally single-user systems as opposed to multi-user systems. "Personal" or "desktop" computers.

non-printing character = A control character which can cause a special function to happen, but does not appear on printed copy. It may or may not appear on the screen. See control character.

number system = In computer usage, several number systems are in use and differ principally in their base. The most commonly used systems are Decimal (base 10), Hexadecimal (base 16), and Octal (base 8). Special print symbols not found on the keyboard listed in most printer manuals, along with the codes required to generate them. MSCRIPT requires the use of Decimal codes for these symbols.

operating system = The control system of the computer under which all software operates. The disk operating system (DOS) is what makes MSCRIPT function as a word processor, and controls the computer and printer. Some popular operating systems are DOSPLUS, CP/M, MS-DOS.

output = (1) Work produced by an operator. (2) Data which is transferred from the computer to a printer, disk drive, etc.

overflow = A condition in which an attempt has been made to store more data in memory than space allows.

overprint = Printing over a character that has previously been printed on the page. This is used for underlining, bold face, foreign accents, strikeouts, etc.

overtype = Moving the cursor over an existing character and typing another, thereby replacing it.

page = (1) The number of lines of text that will be printed on a sheet. (2) One screen of text.

paper out indicator = A light which turns on to inform the operator the printer is placing characters on the platen.

peripheral = A device associated with a computer and connected by interfacing hardware and/or software, such as printers, disk drives, keyboards, etc.

pin-feed = A type of printer which advances continuous forms by means of sprockets engaging holes on either side of the sheet.

pitch The number of characters to be printed in one inch of space on a line, usually 10, 12, or 15. If

proportional spaced printing is used, pitch is meaningless.

platen = The roller or plate against which the printing element strikes the ribbon and paper to produce readable copy.

portable computer = A computer small and light enough to be carried easily from one location to another.

portability = The ability of a program to be used on more than one type of computer.

powerful software = Software that is efficient and provides the user with a wide range of options, particularly if those options are able to be executed with little effort on the part of the operator.

printer = An output device that produces hard copy, or printout.

program = A series of instructions to make the computer function in a certain manner. Software.

prompt = A symbol, word or phrase displayed on the screen to inform the operator of the necessity for some particular action, or to indicate the location of the current character.

proportional spacing = The ability of a printer to move the print head by tiny increments of space rather than a fixed space regardless of the width of the character.

QWERTY keyboard = The standard typewriter keyboard, which is used on most microcomputers.

RAM = Random Access Memory. The memory that is available for programs and data. The "size" of a computer is spoken of in terms of the amount of RAM that is built in. A 96K machine has approximately 96 thousand bytes of memory available. Of the total amount, some is used by the operating system, some by the program(s), and the rest is available for data manipulation.

ROM = Read Only Memory. Memory that is permanent and unchangeable, usually used to perform specific tasks such as printer control, video display, etc.

return = Carriage return; the movement of the print head to the beginning of the line. The Enter key. In this manual, when a press of the return key is indicated, it is displayed this way: <enter>.

reverse video = Display of dark characters against a light background for emphasis.

rewrite = Updating a document, or making changes.

ribbon catridge = The plastic case which houses the print ribbon (carbon film, nylon, etc.) and mechanism for feeding and rewinding.

RS-232 or RS-232C = A widely used standard for interfacing devices by means of a 25-pin connector. The serial I/O port.

scratchpad = An area of memory used as a temporary work area. The buffer.

screen size = (1) The dimensions of the display, usually measured diagonally in inches. (2) The designation of the number of columns and rows displayed on the screen. 80 x 24 and 64 x 16 are common screen sizes.

scrolling = Moving text vertically or horizontally on the screen to enable the operator to view those portions which are not otherwise displayed. The text can be considered a sheet longer and wider than the screen, with the screen serving as a window for viewing.

search and replace = See global find and change.

smart peripheral = A peripheral device such as a printer which contains its own processor and memory, relieving the computer of some of the funcitions. Smart peripherals are usually faster than "dumb" peripherals.

spelling checker = A program that proofreads a document and compares each word with one stored in its dictionary. It marks incorrect words, or words it does not recognize, and allows the operator to correct them. Electric Webster is an example.

storage = Devices and/or media used for holding data

that may be recalled for use at a later date. Hard disks, floppy disks and cassette tapes are forms of storage. Storage is a permanent collection of data, as opposed to memory, which is temporary.

string = A group of characters.

subscript = Characters printed below the normal line of type, as in this example: $_{\text{sub}}$script.

superscript = Characters printed above the normal line, as in this example: $^{\text{super}}$script

tab = Moving the cursor or print head to a column previously determined on the screen or paper.

text entry = The act of typing the document into the computer.

thermal printer = A small, non-impact printer that forms characters by applying heat to specially treated paper.

thimble printer = A letter-quality printer which utilizes a print element that resembles a large thimble. The type elements are at the ends of arms as in a daisy wheel printer, except they are bent upwards from the hub.

tractor feed = See pin feed.

turnkey system = A computer system which contains all the peripherals and programs necessary to perform a particular task such as word processing.

type ball = A molded printer element shaped like a golf ball with raised character elements around the suface. Ball type printers are usually slower than daisy wheel printers because of the more complex movements necessary to position the character to be printed.

user defined keys Special keys on the keyboard that perform certain functions within a program. MSCRIPT allows all ten numeric keys (0-9) to be defined as special printer codes.

user-friendly = Software and hardware systems that are easy for the operator to learn and use without a long period of training.

vertical scrolling = The ability of MSCRIPT to shift the text up and down on the screen to enable the operator to view more text than is visible through the window.

video monitor = A display unit resembling a television set but without a speaker or components for receiving TV signals. A monitor has better resolution of characters than a TV set.

Winchester disk system = A type of hard disk system which was originally developed by IBM, utilizing both sides of the non-removable platters. The name is a play on words, derived from the first production model which was capable of storing 30 megabytes (30,000,000 characters) on each side of the disk, or a 30/30 disk drive.

window = The screen opening which allows the operator to view a portion of the document.

word processing = A computerized solution to the problems created by the writing, editing, revising, formatting, manipulating, storage and printing of documents such as letters, reports, manuscripts.

word wrap = The automatic shifting of an entire word to the next line if it is too long to fit the current line.

write protect = The procedure used to prevent a disk or tape from being inadvertently overwritten by new data. With floppy disks this is accomplished by use of a protective tab being place over the write-protect notch (5.25" disks) or being removed, thus exposing the notch (8" disks); with hard disks it is accomplished by a switch; tape systems employ a plastic ring or tab. Most disk systems provide for write protection by means of software control.

TIMEX 2068
MSCRIPT Addenda

This addenda will cover some peculiarities of MSCRIPT on the TIMEX 2068 which are not otherwise documented.

Command Key

When using the <command> key it is advisable to press the CAPS SHIFT key BEFORE pressing the BREAK key. This will avoid generating an extra space which occurs if this order is reversed.

TAB Feature

In order to use the TAB feature, you must press <command> + Right Arrow.

SCREEN Scroll

In order to use the Scroll by Screen Page feature use the <command> + Up Arrow for UP PAGE, and <command> + Down Arrow for DOWN PAGE.

Print from Cursor

This feature is not available in the TIMEX 2068 version of MSCRIPT.

Saving and Loading files

Always enter a file name in response to a SAVE or LOAD command. MSCRIPT cannot reload a file saved without a name. Please ignore any text in the manual which states otherwise. You must ALWAYS specify a name!! Also MSCRIPT will not search a tape for a proper filename. If the first file it encounters does not have the name of the file that you wish loaded MSCRIPT will return an I/O error. You must either have one file per tape, or using your tape counter, mark the position of each file on your tape.

Instructions for using MSCRIPT files with MTERM (Smart Terminal II)

1) Save your MSCRIPT files to tape as normal. (You should end lines with a <CR> at least every 255 characters. By setting the line length (window) to one character less than your desired final width, you can go to the beginning of each line, press <ENTER> and easily add the required <CR>)

2) Remove your MSCRIPT tape, and then on a blank tape, start recording, and type:

SAVE "text"CODE 26710,20000 <ENTER>

Remove this tape as soon as the header has been recorded. If you listen to the tape, the header is the first bit of noise before the actual bulk of the program. You will hear a tone, then a pause and then the program will follow. We are only concerned with this first bit (the header) and you can stop the tape once this is recorded. You have just made a "Dummy Header" which you will use to fool MTERM into loading your text file into the buffer.

3a) LOAD MTERM, but DO NOT type: PRINT USR 54016.

3b) Put your MSCRIPT file tape back into the recorder and listening to the tape, play it just past the header. When the header has passed stop the tape immediately. Turn the right hub clockwise one half turn and then remove.

4) Place the dummy header tape generated in STEP 2 into your recorder and type:

LOAD "" <ENTER>

Play the dummy header tape and stop it soon as the header is read, and then put the MSCRIPT file tape in and PLAY it. Your text file should now load into the buffer. This load will stop with an error message, that's OK – it does not mean that there is a problem, this error occurs because your file is more than likely shorter than 20000 bytes which the dummy header says should be there.)

5) Type: POKE 23628,200

6) Type: PRINT USR 54016.

You now have your file in the buffer. And you can transmit it to another MSCRIPT owner with a 2050 Modem or a BBS System.

– Mark L. Fendrick –

TYPESTYLES and Other Tricks on the Memotech DMX–80, Panasonic 1090,91 Printers

To change your Typestyle you must enter an imbedded code into your text. To do this follow these steps exactly:

1.Define your code keys in the form:

>(0–9)=XX,

These definitions should go at the beginning of your text; (think of 0–9 as macrokeys ;XX as the number from below). You can have more than one definition per line as long as you separate them by commas and end the whole thing by a new line marker (which is the back slash on the 2068).

2.Whenever you want a different type style you move your cursor to one space before where you want your new type style to begin and press a <funct>–G (Symbol Shift + Caps Shift + G). This will print a small copyright symbol on the screen and give you one blank space. In this newly opened up space enter the code number of the macrokey of the type style you want from the tables below.

REMEMBER THIS FORMAT; >#n=XX
(Where n = 0–9)

CODE	WHAT IT DOES
14....	Double Width print... Released by line feed.
15...	Compressed print.. Not to be confused with type styles.
18...	Releases compressed print.
27+52.	Turns on Italics print.
27+53.	Turns off Italics print.
27+80.	Pica type style.
27+77.	Elite type style.
27+111.	Proportional type style.
+ and –	These are the codes for emphasizing and underlining. Enter these codes to underline or emphasize text.

REMEMBER: Type styles are toggles there is no need to switch them off, just switch from one to the other.

NOW FOR SOME EXAMPLES;

THIS IS ENLARGED PRINT
The code of enlarged print is 14

THIS IS COMPRESSED UPPER and lower case
The code to turn compressed print on is 15
The code to turn compressed print off is 18

THIS IS ITALICS UPPER and lower case
The code to turn Italics on is 27+52
The code to turn Italics off is 27+53

THIS IS PICA TYPE UPPER and lower case.
The code for Pica is 27+80

THIS IS ELITE UPPER and elite lower case.
The code for elite is 27+77

THIS IS PROPORTIONAL UPPER and lower case.
The code for proportional is 27+111

This text is underlined...

This text is printed in BOLD TYPE!

By taking advantage of the different print and type styles that your printer offers you can make letters, documentation, and reports look and read better.

Index

Append 23
Backup 12, 51
Block 7
Block Copy 19
Block Delete 20
Bold Face 32
Buffer 7
Bulletins 50
Character 7
Column 8
Columnar Text 48
Command Menu 9
Comment Lines 35
Cursor 6
Delete Block 20
Delete Character 14
Delete Multiple Characters 16
DOS Commands 26
Edit Menu 14
Editing 7
Envelopes 45
Flush Right Text 33
Footers 37
Footer Margins 39
Forcing Page End 40
Form Letters 47
Format Line 27
Global Change 22
Global Find 20, 43
Glossary 55
Hanging Indent 33
Headers 37
Header Margins 39
Help Menu 14
Imbedded Commands 35
Insert Character 15
Justification 31
Letterheads 45
Line Length 31
Load 7
Mail Lists 45
Margins, Setting 31
Marking a Block 19
Menu 6
Non-Printing Lines 35
Overtyping 14
Page End 40
Page Numbers 40
Paragraph 7
Print Menu 27

Index

Printer Codes 35
Printing a Document 26
Programs 50
Saving a Document 7, 25
Sentence 7
Setting Tabs 24
Special Applications 43
String 7
Tabs 24
Text Entry 7
Underlining 32
Update 7
Video Line 6
Window 7
Window, Adjusting Width 24
Word 7

MSCRIPT Command Summary

FORMAT LINE COMMANDS:

>JU=Y Justify On
>JU=N Justify Off
>CE=Y Center Text
>CE=N End Centering
>LM=xx Set Left Margin
>LL=xx Set Line Length
>LS=xx Set Line Spacing
>FR=Y Set Flush Right
>FR=N End Flush Right
>PN=xx Beginning Page no.
>HI=Y Start Hanging Indent
>HI=N Return to Normal Indent
>#n=x/x Define Print Code (n = 0–9)

HEADERS:

>TT=/x/x/x/ Prints on All pages.
>OT=/x/x/x/ Prints on Odd pages.
>ET=/x/x/x/ Prints on Even pages.

FOOTERS:

>BT=/x/x/x/ Prints on All pages.
>OB=/x/x/x/ Prints on Odd pages.
>EB=/x/x/x/ Prints on Even pages.

Position between slashes indicates flush left, center and flush right positions for Header and Footer Text. Use "$" for page no.

TEXT ENTRY/EDIT COMMANDS:

<command> + → : Moves to next Tab Stop.
<command> + ↑ : Scrolls cursor up a Video Screen.
<command> + ↓ : Scrolls cursor down a Video Screen.
BOLD FACE Begin/End: <funct>–G and +.
UNDERLINING Begin/End: <funct>–G and –.
IMBED PRINTER CODES: <funct>–G followed by number of code (1–9).
HELP SCREEN: <funct>–H

MSCRIPT Command Summary

INSERTING:

Open Insert Block: <funct>–I
Merge text: <funct>–M
Moving the cursor off the line merges, leaving a space.

DELETING:

Delete a Character: <funct>–D
Delete a Word: <funct>–SW
Hack to End of line: <funct>–SH
Remove spaces to next character: <funct>–S<spacebar>

BLOCK COMMANDS:

Mark Block Begin/End: <funct>–B
Copy Block to cursor position: <funct>–C
Move Block to cursor position: <funct>–R
Unmark all Blocks: <funct>–U
<funct>–C and <funct>–R act on the first block only, but <funct>–U unmarks all blocks simultaneously.

GLOBAL FIND AND CHANGE:

At Command Menu enter string to be found with:
Fstring
To find the next occurance use <funct>–F. To change every Find string use:
Cstring.

SETTING TABS

At Command Menu, Tx,x,x,x sets Tab Stops. Pressing <funct>–Left Arrow at Text Entry/Edit moves cursor to next tab stop.